★ More Original Programs for your Apple //! ★

Inside — A complete, new Apple language!

$7.30

nibble®

THE REFERENCE FOR APPLE COMPUTING

January 1986
VOL. 7 / NO. 1
U.S.A. $3.25
CANADA $4.25

Turtle Basic
Logo power from Applesoft

Nibble Calculator
It's programmable

Nibble Light Pen
Make your own
for under $10

Tank Combat
Hi-Res action and
strategy combined

Plus ProDOS utilities,
assembly language explained,
games reviewed, and more!

01
0 74470 12699

Let's compare Apples™ to Apples™.

An Apple IIc

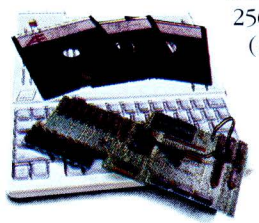
An Apple IIc with Z-RAM

The Apple IIc on the right works exactly the same as the Apple IIc on the left. Almost. The Apple on the right has a powerful memory expansion coprocessing card called Z-RAM. From Applied Engineering. Which means the Apple on the right can completely load AppleWorks into RAM—and then run it up to thirty times faster than the Apple on the left.

Z-RAM also acts as a solid-state disk drive. Which means the Apple on the right will load and store programs up to 30 times faster. And, our included RAM disk is compatible with Applesoft, PRO-DOS, DOS 3.3, PASCAL and CP/M.

Turbo Charged AppleWorks.

Even a 256K Z-RAM can completely load AppleWorks into RAM. With Z-RAM, the moment your fingers touch the keyboard AppleWorks responds. A 256K Z-RAM lets your IIc run AppleWorks up to 30 times faster, increases available desktop to 235K and maximum number of records from 1,350 to over 16,000, doubles the number of lines allowed in the word processor, provides a print spooler, and auto-segments large files so they can be saved on two or more disks. A 512K Z-RAM boosts AppleWorks desktop to an incredible 425K.



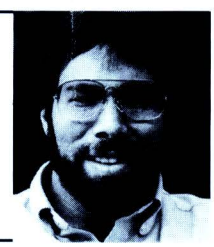
Take a closer look.

There's more. Z-RAM has a built-in high speed Z-80B microprocessor that allows you to run CP/M programs. Which means you now have access to the single largest body of software in existence, including popular packages like WordStar, dBase II, Turbo PASCAL and Microsoft BASIC. A 16 bit option is also available.

And still more. Z-RAM is compatible with all IIc software and hardware, installs easily in just ten minutes with a screwdriver (slightly longer without), is available with 256K or 512K of additional memory (a 256K Z-RAM can be upgraded to 512K at any time). Z-RAM is easily handled by the IIc power supply with our patent pending power saving design.

The only thing better than that would be a recommendation from Steve Wozniak.

> ***"I recommend Applied Engineering products whole-heartedly." (Of course, Steve's IIc has a Z-RAM installed.)***
>
> *Steve Wozniak, the creator of Apple Computer*



For fast response.

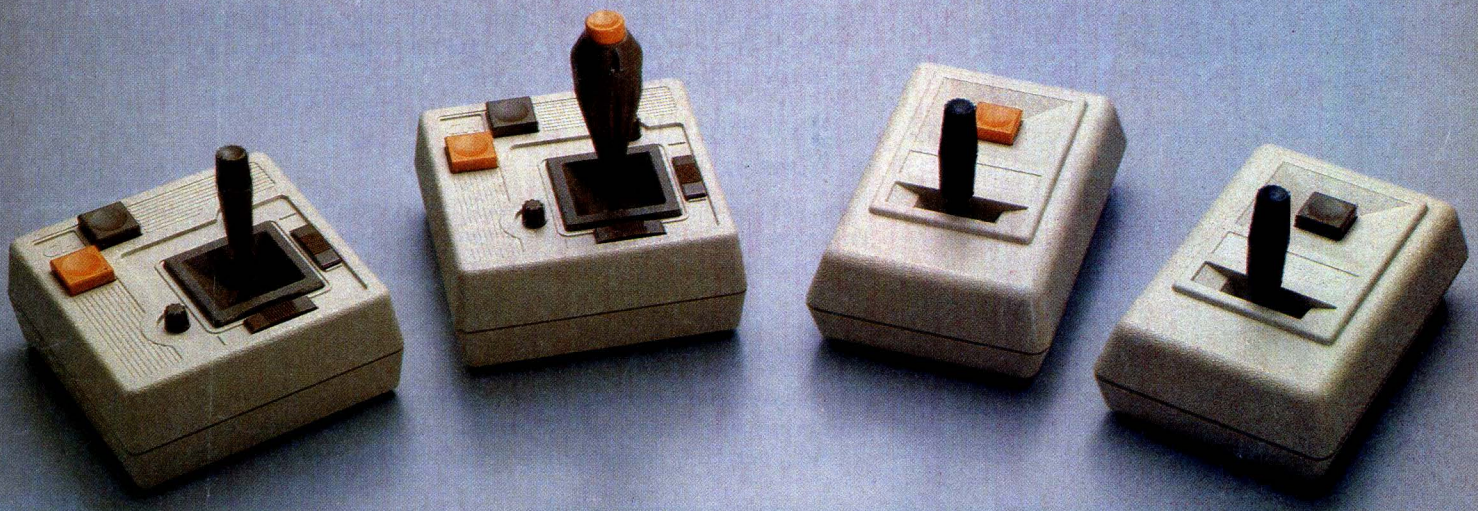
Z-RAM comes complete with simple instructions, RAM disk software, Z-80 operating system, CP/M manual. And a five year "hassle free" warranty. Make a good Apple great. With 256K Z-RAM "384K total" ($359); with 512K "640K total" ($419); 16 bit option may be added later ($89).

If you want to run CP/M software, but don't need more memory, we suggest our Z-80c card. The Z-80c has no memory expansion ports and is priced at only $159.

Call 214-241-6060 TO ORDER TODAY. 9 a.m. to 11 p.m. Seven days, or send check or money order to Applied Engineering. MasterCard, Visa and C.O.D. welcome. Texas residents add 5⅛% sales tax. Add $10.00 outside U.S.A.

AE Applied Engineering
P.O. Box 798, Carrollton, TX 75006
(214) 241-6060

Superior firepower



►MACH II™ and MACH III™ JOYSTICKS

The second generation MACH II and MACH III Joysticks incorporate a new slide-switch spring disconnect feature that requires no stick deflection and new rotary trims which are four times more precise.

Additional features include: "spring centering" or "positive true-positioning" modes of operation, electrical trims for each axis, two fire buttons on the deck of the control and a stainless steel ball as the main pivot for added precision and durability. Our MACH III Joystick has an added fire button on the stick handle.

These highly accurate and reliable 360-degree microcomputer cursor controls used for games, business and graphics applications provide more features, greater durability, and longer life cycles. Plug-in compatible with Apple II, IIE, IIC, IBM PC, PCjr and TRS-80 Color Computers.
MACH II **$44.95** / MACH III **$54.95**

►PADDLESTICKS™

PADDLESTICKS are a new and unique paddle control design that promises to return the fun and excitement back to paddletype games. They are faster, easier to operate and less fatiguing than conventional-type paddles. The PADDLESTICKS operate as a single-axis joystick and are currently plug-in compatible with the Apple II, IIE and IIC personal computers.
$44.95 (pair)

We manufacture a complete line of OEM Joysticks for industrial applications. Dealer and OEM inquiries invited. Apple II, IIE, IIC, IBM PC, PCjr and TRS-80 are registered trademarks of Apple Computer Inc., International Business Machines and Tandy Corp., respectively.

CH products™

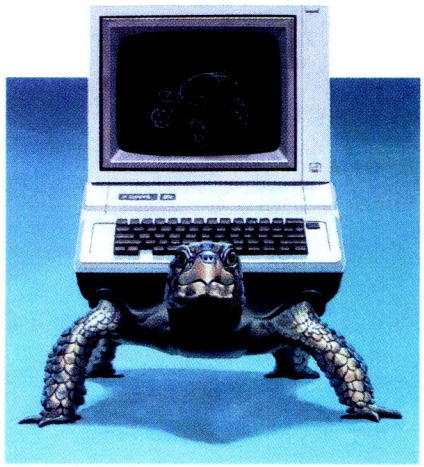
Formerly Hayes Products
1558 Osage Street
San Marcos, California 92069
(619) 744-8546

More bang for the buck.

Manufactured in U.S.A.

Volume 7/Number 1
January 1986

nibble®



cover art by Don Demers

Page 14

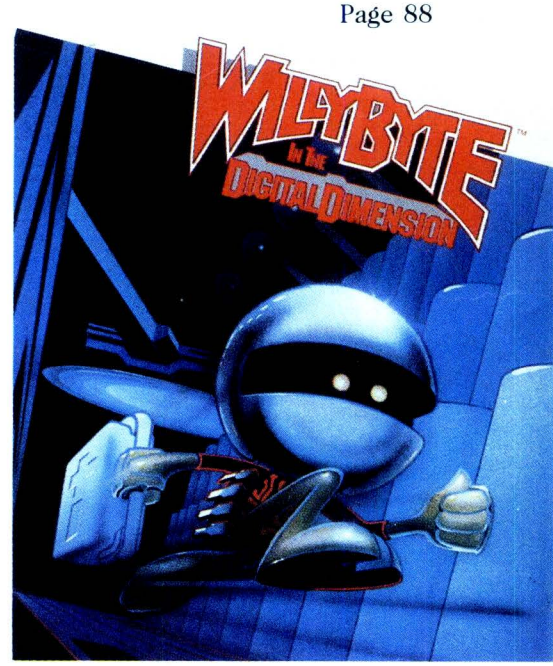
FEATURES

TIPS 'N TECHNIQUES

APPLE UTILITIES

USER VIEWS

Page 88



NIBBLE Magazine (ISSN 0734-3795) is published monthly for $3.25 per issue, $26.95 for a one-year subscription (Canada surface $34.95, outside U.S. and Canada surface $39.95) by MicroSPARC, Inc., 45 Winthrop Street, Concord, MA 01742. Second-Class postage paid at Concord, MA and at additional mailing offices. POSTMASTER: Send address changes to NIBBLE Magazine, 45 Winthrop St., Concord, MA 01742.

Address all editorial, advertising and subscription inquiries to NIBBLE's Concord offices. Return postage must accompany all manuscripts, drawings and photographs submitted if they are to be returned, and no responsibility can be assumed for unsolicited materials. All letters sent to NIBBLE will be treated as unconditionally assigned for publication and copyright purposes and as subject to NIBBLE's right to edit and comment editorially. All rights reserved. Nothing may be reprinted in part or in whole without written permission from the publisher. The editorial board assumes no liability or responsibility for the products advertised in this magazine.

NIBBLE will replace lost copies for 60 days following the publication date. Beyond 60 days, we cannot be responsible for mail loss.

The Reference for Apple Computing

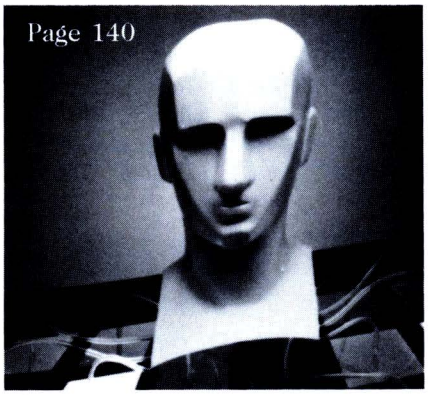
NIBBLING AT ASSEMBLY LANGUAGE

TIPS 'N TECHNIQUES

DISASSEMBLY LINES

HARDWARE CONSTRUCTION PROJECT

Page 56



Page 140

DEPARTMENTS

In Germany, dealer inquiries should be directed to: International Presse, Borsigalle 17, D6000 Frankfurt, 60 WG. Tel: (069) 419 198.

or:

Pandasoft, Uhlandstr 195, 1000 Berlin 12, Germany. Tel: (030) 310 423.

THE X-10 POWERHOUSE DOES EVERYTHING BUT PUT OUT THE CAT.

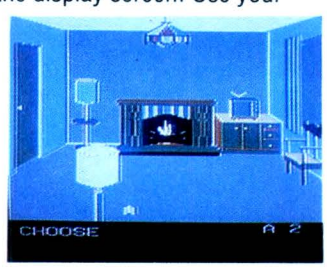
Model CP290

THE X-10 POWERHOUSE INTERFACES WITH YOUR APPLE TO CONTROL YOUR HOME...FOR SECURITY, COMFORT AND ENERGY SAVINGS.

This remarkable Interface lets you run your home through your Apple IIe or IIc and a mouse, keyboard or joystick.

When you're away, it makes your home look and sound lived in. When you're home, it can turn off the TV at night and wake you up to stereo and fresh brewed coffee in the morning. It can even turn on your air conditioner and control your heating.

SPECIAL COLOR GRAPHICS MAKE PROGRAMMING A SNAP. You simply pick a room from the display screen. Use your mouse, joystick or keyboard to position graphics of lights or appliances. Then follow on-screen instructions to program any light or appliance to go on or off whenever you choose. You can even control thermostats, light intensity and more.



THE WAY IT WORKS. The X-10 Powerhouse Interface is cable-connected to the Apple RS-232 port and plugged into a standard 110V outlet. After it is programmed, the Interface sends digitally encoded signals through your home wiring to special X-10 Modules. To control a lamp or appliance, you simply plug the electrical device into a Module and then plug the Module

into an outlet. The Interface can control up to 256 Modules throughout your home and won't interfere with normal use of lights and appliances.
There are plug-in Appliance Modules, Lamp Modules, Wall Switch Replacement Modules and Special 220V Modules for heavy duty appliances such as water heaters and room air conditioners. Plus Thermostat Controllers for central heating and air conditioning, Telephone Responders to control your home from any phone, and much more.

IT WON'T TIE UP YOUR COMPUTER. Use your computer only for programming. When you're finished, disconnect the Interface from the RS-232 port and keep it plugged into any convenient power outlet in your home. It will operate as a stand-alone controller with battery back-up and will run your home automatically.

SURPRISINGLY INEXPENSIVE. A Powerhouse System including the Interface, software and connecting cables costs less than $150. X-10 Modules are less than $20 each.

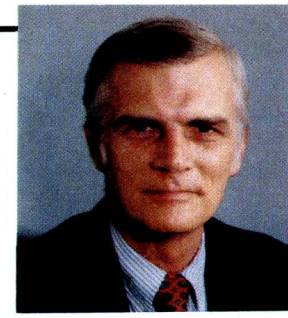
For the Dealer Nearest You Call: 1-800 526-0027
[in NJ: (201) 784-9700]
or, write to: X-10 (USA)
185A Legrand Avenue
Northvale, NJ 07647

X-10® POWERHOUSE
NUMBER ONE IN HOME CONTROL

Apple, Apple IIe and Apple IIc are registered trademarks of Apple Computer, Inc.

CIRCLE NUMBER 1

ELEPHANTS AND ANTELOPES



VIEWPOINT

With this issue, we begin our seventh year of publication. We've also reached a major milestone: with this issue, we expect to exceed 100,000 immediate circulation for the first time in *Nibble's* history. Thank you for helping to make it happen! Your encouragement and support have kept us going when other, far larger magazines have foundered and failed. We were particularly saddened to see the demise of Creative Computing and Popular Computing this past year.

ELEPHANTS

Elephants have big appetites. That applies to businesses too. One of the unwritten business guidelines that we used to hear about when I worked for Xerox was: "We're not interested in any business that produces less than $100 million of annual revenue."

In a recent interview, David Ahl claimed that he projected a million dollar profit for Creative Computing a short time before the decision was made to cease publication. He said that he wasn't sure whether the decision makers didn't believe him or thought it wasn't enough.

When the decision was made to kill Popular Computing, the staff had just moved into a new $1.2 million headquarters building, and the magazine claimed a paid circulation of more than 300,000. It either wasn't big enough or wasn't profitable enough to survive.

In the last several years, more than 50 computer magazines have ceased publication. The battlefield is littered with big names. It's happening in software as well.

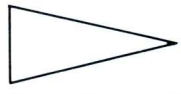
ANTELOPES

Antelopes are fast, little creatures. They don't eat a lot — at least not as much as elephants. They have no excess bulk. They are survivors. A company that's an antelope is delighted to sell 3,000-5,000 software packages a month. An elephant thinks that kind of performance spells failure.

Stewart Alsop, formerly with InfoWorld, made an insightful comparison between Lotus and MicroSoft. He pointed out that Lotus hasn't figured out how to make money with small products. MicroSoft has. What Lotus does with six products (and really with 1-2-3 as the predominant product), MicroSoft does with more than 30 products. Lotus is looking more and more like an elephant, while MicroSoft appears to be a herd of antelopes.

THE OPPORTUNITY WEDGE

A classic management model states that OPPORTUNITIES ARE CONTINUOUSLY CLOSING. It applies to a wide variety of business and personal experiences, and it can be represented as a wedge like this:



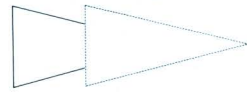
Point A is the beginning of an opportunity. When VisiCorp popularized the spreadsheet, there were no spreadsheets on the market. The opportunity was huge, and VisiCorp capitalized on it with almost no competition. But the opportunity began to shrink.

At point B, the opportunity is maturing. Competition, market saturation, product obsolescence, bad pricing, and a myriad of other market forces enter into the picture.

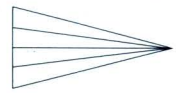
Point C is the disappearance of the opportunity. At this point the profit or market is no longer big enough for survival.

NEW WEDGES

Of course, the wedge can be made to move by introducing new technology, creativity and market product development.



The double wedge defines a Lotus-like phenomenon with one product dominating the financial/market scene. It remains to be seen whether the second wedge will be of Lotus' creation or whether it will be created by another company (repeating the VisiCalc phenomenon).



This wedge defines an equally large opportunity that is constructed of many smaller, more diverse opportunities. It is most representative of companies like MicroSoft and a host of other smaller companies that survive and thrive on a variety of small enterprises.

NIBBLE...MANY WEDGES

Nibble is made up of many wedges. Circulation, advertising and more than 150 software products all fit into a supportive relationship that keeps the lights burning. This strategy of controlled, diversified growth has enabled us to survive and prosper while a lot of elephants have starved.

Several years ago, we had two opportunities to be acquired by much larger publishing companies. The offers were tempting, since they would have made my family financially independent for life. And an acquisition held the promise of a major infusion of resources that would make *Nibble* the biggest Apple magazine in the market.

Today, I am convinced that if we had allowed ourselves to be acquired, we would no longer be in business. The weekend that I had to reach a decision, I went out into the yard, sat under a tree and thought about priorities. I thought about you, our readers and advertisers. I thought about our employees — some of the finest in the business. And I thought about how elephants live and die.

As we enter our seventh year of publication, let me thank you once again for helping us keep the antelope spirit and vitality.

Mike Harvey

Mike Harvey
Publisher

nibble®

PUBLISHER — Mike Harvey
EDITOR — David Szetela
MANAGING EDITOR — David A. Krathwohl

EDITORIAL
Technical Editors — Loren Wright, Rich Williams
Copy Editors — Sharon Stentiford, Carol McGarry
Technical Assistant — Steven Sarsfield
Executive Secretary — Iris Meyerkopf

ADMINISTRATION
Asst. to the Publisher — Adele P. Harvey
Controller — Diane Carhart
Accounts Receivable — William Lawler
Accounts Payable — Mara Gurski
Admin. Assistant — Siobhan O'Brien
Dealer Administration — David Gasdia

DESIGN/PRODUCTION
Production Manager — Paul J. Gagnon
Production Coordinator — Rebecca Richardson
Typesetting — Victorine Belanger

ADVERTISING SALES
National Sales Manager — Kevin Rushalko
Account Executive — Andrew Mintz

ADVERTISING ADMINISTRATION
Manager — Charlotte W. Szetela
Assistant — Sally Abu-moustafa

DEALER/DISTRIBUTOR MARKET DEVELOPMENT
Manager — Cathi Bland

CIRCULATION/ORDER PROCESSING
Manager — Kelley Magiera
Assistants — Karyn Mohen, Lee Perry

CUSTOMER SERVICE
Coordinator — Debbie Levins

SHIPPING & INTERNAL SERVICES
Manager — David Sears
Assistants — William G. Brown, Randall Raja

MicroSPARC Inc.
President and C.E.O. — Mike Harvey
Vice President — Adele Harvey
Vice President, Publishing Operations — David Szetela
Vice President, Finance — Diane Carhart

LETTERS

Foolproof Exhumation

I think I've discovered a bug in Sandy Mossberg's EXHUME.FILE program ("Recovering Deleted ProDOS Files," Vol. 6/No. 10). If I rename a file to a shorter file name (e.g., RENAME LONGFILE,LONG) and then delete the file, EXHUME.FILE is unable to recover it. Also, EXHUME.FILE can't restore a file once the disk has been renamed using FILER. What gives?

Gino Fortunato
Baton Rouge, LA

Sandy Mossberg replies:

Right you are, Gino. My shortsightedness about renamed files has created unneeded frustration. After expanding upon your explanation for the cause of the malfunction, I shall provide a short fix.

Within each file header or entry, the file name field (FNF) consists of 15 bytes. The low order nibble of the preceding byte contains the length of the file name (NLF). When a file is created, its name and name length are placed within the FNF and NLF, respectively.

LISTING 1: TANK.COMBAT *(continued)*

```
3220  FOR I = 8 TO 1 STEP  - 1: FOR J = 1 TO
      18: GOSUB 3530
3230  IF N = 2 AND A$(I,J) = "" THEN A$(I,J) =
      "M1":FL = FL + 1: IF FL = 6 THEN I = 1:J
       = 18
3240  NEXT J,I: IF FL < 6 THEN 3220
3250 FL = 0
3260  FOR I = 3 TO 1 STEP  - 1: FOR J = 1 TO
      18: GOSUB 3530
3270  IF N = 1 AND A$(I,J) = "" THEN FL = FL +
      1:A$(I,J) =  STR$ (FL) + "C":XC(FL) = I:
      YC(FL) = J: IF FL = 6 THEN I = 1:J = 18
3280  NEXT J,I: IF FL < 6 THEN 3260
3290  FOR I = 1 TO 3: FOR J = 1 TO 18
3300  IF  VAL (A$(I,J)) > 0 THEN  XDRAW 36 AT
      J * 14,Y + (I * 9)
3310  NEXT J,I
3320  REM  HUMAN LOCATIONS
3330  POKE 34,20: HOME
3340  REM  PLACE HQ
3350 OB$ = "HQ": GOSUB 4120:RS = 16:RE = 9:R =
      16:C = 1:SH = 32:A$ = "K": GOSUB 3710
3360 A$(R,C) = "HQ2"
3370 OB$ = "AMMO DUMP": GOSUB 4120:RS = 16:RE
       = 9:R = 16:C = 1:SH = 33:A$ = "K": GOSUB
      3710
3380 A$(R,C) = "AD2"
3390 OB$ = "FUEL DEPOT": GOSUB 4120:RS = 16:R
      E = 9:R = 16:C = 1:SH = 34:A$ = "K": GOSUB
      3710
3400 A$(R,C) = "FD2"
3410  FOR K = 1 TO 6
3420 OB$ = "TANK MINE #" +  STR$ (K): GOSUB 4
      120:RS = 16:RE = 9:R = 16:C = 1:SH = 19:
      A$ = "K": GOSUB 3710
3430 A$(R,C) = "MI2"
3440  NEXT
3450  FOR I = 1 TO 6:C = 1:OB$ = "TANK #" +  STR$
      (I): GOSUB 4120
3460 RS = 16:RE = 14:R = 16:C = 1:SH = 31:A$ =
      "K": GOSUB 3710
3470 A$(R,C) =  STR$ (I) + "H":XH(I) = R:YH(I
      ) = C
3480  NEXT
3490  FOR I = 9 TO 16: FOR J = 1 TO 18
3500  IF A$(I,J) = "MI2" THEN  XDRAW 19 AT J *
      14,Y + (I * 9)
3510  NEXT J,I
3520  RETURN
3530  REM  PICK RND NO.
3540 N =  INT ( RND ( PEEK (78) +  PEEK (79) *
      256) * NU) + 1
3550  RETURN
3560  REM  DRAW AND ERASE TANK MINE
3570  FOR D = 1 TO 10: XDRAW 19 AT C * 14,Y +
      (R * 9): FOR E = 1 TO 50: NEXT : NEXT : RETURN

3580  REM  SHOW WHERE SHOT LANDED (COMPUTER)
3590  CALL 768
3600  FOR D = 1 TO 20: XDRAW 35 AT CY(I) * 14
      ,Y + (CX(I) * 9): FOR E = 1 TO 50: NEXT
      E,D: RETURN
3610  REM  SHOW WHERE SHOT LANDED (HUMAN)
3620  CALL 768
3630  FOR D = 1 TO 20: XDRAW 35 AT HY(I) * 14
      ,Y + (HX(I) * 9): FOR E = 1 TO 50: NEXT
      E,D: RETURN
3640  FOR D = 1 TO 3000: NEXT : RETURN
3650  HTAB 21 -  LEN (M$) / 2: PRINT M$: RETURN

3660  FOR X = 768 TO 781
3670  READ Y: POKE X,Y: NEXT
3680  RETURN
3690  DATA 160,20,169,50,32,168,252,173,48,19
      2,136,208,245,96
3700  REM  MOVE SHAPES ON SCREEN USING I-J-K-
      M AND <SPACE>.
3710  IF SH = 37 THEN  GOSUB 4050: GOTO 3790
3720  IF SH = 31 THEN  IF A$(R,C) = "B" OR  RIGHT$
      (A$(R,C),1) = "H" OR  RIGHT$ (A$(R,C),1)
       = "C" THEN 3850
3730  IF FL THEN FL = 0: GOTO 3720
3740  IF SH = 19 THEN  IF  LEFT$ (A$(R,C),1) =
      "M" THEN 3850
3750  IF FL THEN FL = 0: GOTO 3740
```

should have provided a program with AppleWorks for easy conversion of Applewriter and other text files to AppleWorks word processor files. Such a conversion program is provided with the database to convert Quickfile data.

The process for converting DOS 3.3 text files, like those created using Applewriter, is not self-evident, involves several steps, and you must own a copy of ProDOS. Here is my method:

1. Write down the names of the DOS 3.3 text files that you wish to convert. Boot the ProDOS User's Disk and choose F (ProDOS Filer), then V (Volume Commands). Next, choose F (Format a volume). Type in and write down the new volume name (the prefix). Remove the ProDOS User's Disk or set the formatting for drive 2, insert a fresh disk and proceed to format it.
2. Replace the ProDOS User's Disk. Go back to the main menu (via Quit or by rebooting) and select C (the conversion option). At this point, if you are unfamiliar with the use of the conversion program, type a question mark to display instructions.
3. Remove the ProDOS User's Disk and insert the DOS 3.3 and ProDOS formatted disks, as shown under "Direction". Be sure that the slot and drive numbers are set correctly in the conversion program (if not, use C to change them). Set the ProDOS prefix using menu option P, and type in the prefix you gave in number 1 above. (Use the <RETURN> key to eliminate any extra characters in the prefix). Check to see that the arrow is set correctly for the direction of the conversion — if it is not correct, use R to correct it.
4. Choose option T to transfer the files from the DOS 3.3 disk to the ProDOS disk and proceed to transfer the files according to the directions you read in step 2 above. If you have forgotten the file names to be transferred, enter a question mark instead of the file names to display a catalog of the disk files. The transfer option converts DOS 3.3 ASCII text files to ProDOS ASCII text files. However, an extra step is required to convert the latter to AppleWorks word processor (WP) files. Remove the DOS 3.3 disk and place the new ProDOS text disk in drive 2.
5. Boot AppleWorks. Choose the "Add Files" option and "Make a new file for the Word Processor." Then choose option 2 for a text (ASCII) file.

continued on page 146

ERRATA AND ENHANCEMENTS

Attention Telecommunicators: Now you can receive errata notices and other goodies quicker than ever on the 1200/300 baud Nibble Hot Line. Have your Apple call our Apple at 617-369-8920. Your communications software should be set at 8 data bits, 1 stop bit, no parity, full duplex. The Hot Line will automatically determine the correct baud rate.

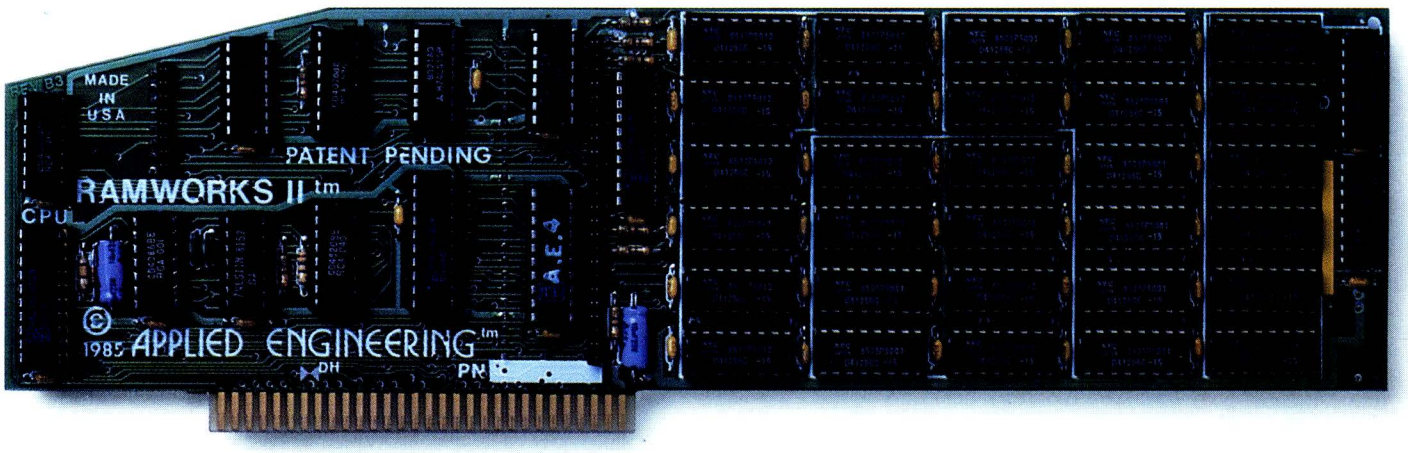
Super Shopper (Vol. 6/No. 11, p. 22): To fix a problem with Epson printers not entering 132-character mode, change line 290 as follows:

```
290 P132$ = CHR$(15)
```

Nibble MeterMan (Vol. 6/No. 10, p. 28): To fix a problem with the printout of the 80-column screen, eliminate the semicolon in line 1890.

Meet RamWorks II®

The Recognized Industry Standard For Memory Expansion of the Apple IIe.



RamWorks II. A Generation Ahead. Again.

The best selling expansion card for the Apple IIe just got even better. With RamWorks II, expand your IIe to an incredible 3 megabytes of usable RAM.

Turbo Charged AppleWorks.

RamWorks II plugs into the IIe auxiliary slot and acts just like Apple's extended 80 column card, only better—because if you buy a 256K or larger card, AppleWorks will automatically load itself into RamWorks II. This dramatically increases AppleWorks' speed and power because it effectively eliminates the time required to access disk drive 1. Now, switch from word processing to spreadsheet to database management at the speed of light. AppleWorks responds the moment your fingers touch the keyboard.

But AppleWorks has certain internal limits, independent of available memory. Fear not. Only RamWorks II (and the original RamWorks of course) removes those limits. Only RamWorks II increases the maximum number of records available from 1,350 to over 16 000. Only RamWorks II actually increases the number of lines permitted in the word processing mode. And only RamWorks II features a built-in printer buffer, so you no longer have to wait for your printer to stop before going back to AppleWorks (256K or larger RamWorks II required).

With RamWorks II, you won't have to split your data into 2 or more separate files because you'll have the necessary memory to access ALL your data ALL the time, quickly and conveniently.

RamWorks II	AppleWorks Desktop
128K	101K
256K	188K
512K	378K
1 MEG	758K
1.5 MEG	1136K
3 MEG	2277K

The Most Friendly, Most Expandable Card Available.

RamWorks II is compatible with more off-the-shelf software than any other RAM card. Popular programs like Advanced VisiCalc, Magic Office System, Flashcalc, The Spread Sheet, Diversi-DOS, Supercalc 3A, Magicalc, etc. (and hardware add-ons like Profile and Sider hard disks). Fact is, only RamWorks is 100% compatible with all software written for the Apple 80 column and extended 80 column cards. In addition, RamWorks II can emulate most other RAM cards, so you can use programs written for them without modification. And any size RamWorks II can be user upgraded later to any larger size.

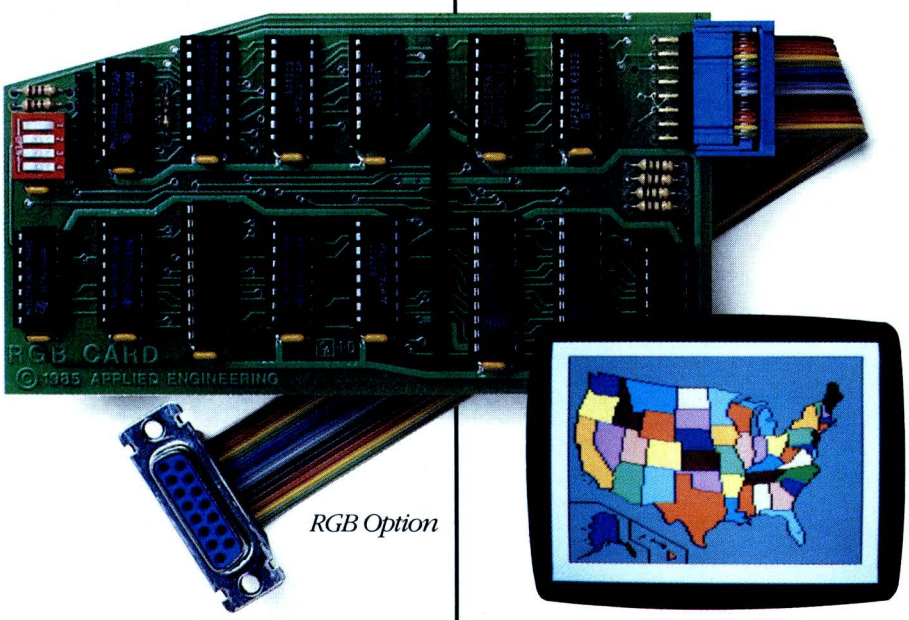
RamWorks II was designed so you could take full advantage of future developments in 16 and 32 bit microprocessors. As your needs grow, so can RamWorks II. A handy coprocessor connector allows the latest and greatest coprocessor cards to access all 3 MEG

of RamWorks II memory. And speaking of more memory, RamWorks II has a memory expansion connector on board so a low profile (no slot 1 interference) memory expansion card can add another 512K or 2 MEG of memory.

Unlike Apple's smaller, more expensive RAM card, RamWorks II plugs into the IIe auxiliary slot and therefore leaves slots 4 and 5 available for other peripheral cards.

It's In Color

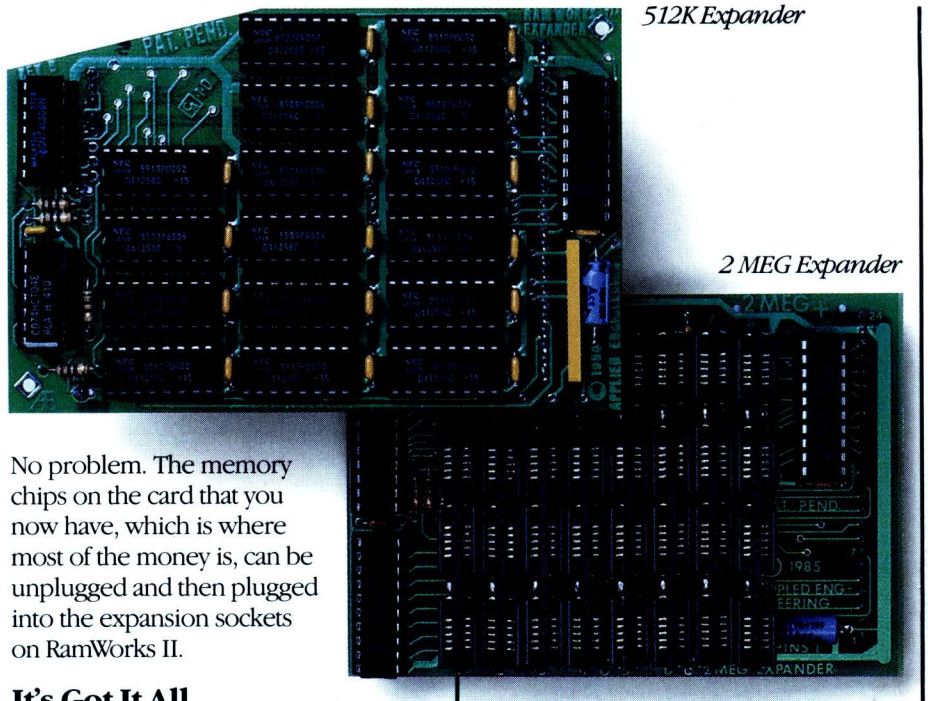
RamWorks II by itself is *fully* compatible with both the Apple monochrome and color monitors. But if you want better color graphics *plus* a more readable 80 column text (that blows away any composite color monitor) you'll appreciate our RGB color option. For only $129, it can be added to RamWorks II, giving you a razor sharp, vivid brilliance that's unsurpassed in the industry. The RGB option does not waste another valuable slot, but rather plugs into the back of RamWorks II with no slot 1 interference (works on the original RamWorks, too) and attaches to virtually any RGB monitor. And remember. You can order the RGB option with your RamWorks II. Or add it on at a later date.



RGB Option

It Corrects Mistakes.

Let's say you bought some other RAM card (and that's a mistake) and your RAM card is not being recognized by AppleWorks, Advanced Visicalc, Flashcalc, Supercalc 3A, or other programs, and you want RamWorks II.



512K Expander

2 MEG Expander

No problem. The memory chips on the card that you now have, which is where most of the money is, can be unplugged and then plugged into the expansion sockets on RamWorks II.

It's Got It All.

- 15 Day Money Back Guarantee
- Super sharp 80 column text (patent pending) with or without RGB option
- Double high resolution graphics (with or without RGB option)
- Expandable up to 1 Meg (1024K) on main board
- Expandable to 3 Meg (3072K) with expander (piggyback) card
- Can use 64K or 256K RAMS in any combination
- Linear addressing coprocessor port
- Automatic AppleWorks expansion up to 2277K desktop
- Accelerates AppleWorks
- Built-in AppleWorks printer buffer
- The only large RAM card that's 100% compatible with all IIe software
- RamDrive™ the ultimate disk emulation software included free
- 16 Bit option
- Compatible RGB option
- Built-in self diagnostics software
- No slot 1 interference
- Lowest power consumption (patent pending)
- Takes only one slot (auxiliary)
- Software industry standard
- Advanced Computer Aided Design
- Used by Apple Computer, Steve Wozniak and virtually all software companies
- Displays date and time on the AppleWorks screen with any PRO-DOS compatible clock
- 5 Year no hassle warranty

RamWorks II with 64K	**$ 179**
RamWorks II with 256K	**$ 219**
RamWorks II with 512K	**$ 269**
RamWorks II with 1 MEG	**$ 389**
RamWorks II with 1.5 MEG	**$ 549**
RamWorks II with 3 MEG	**$1699**
RGB Option (may add later)	**$ 129**
16 Bit Option (may add later)	**$ 89**

RamWorks II. The industry standard for memory expansion of the Apple IIe.

ORDER YOUR RamWorks II TODAY. 9 a.m. to 11 p.m. 7 Days, or send check or money order to Applied Engineering.

MasterCard, Visa and C.O.D. welcome. Texas Residents add 5⅛% sales tax. Add $10.00 outside U.S.A.

Applied Engineering
P.O. Box 798, Carrollton, TX 75006
(214) 241-6060

A WELCOME TO NEW NIBBLE READERS

Welcome to *Nibble*, one of the finest publications available for Apple owners. Each month, *Nibble* publishes an outstanding collection of programs for your Apple along with explanations of how they work.

If you're new to computing, however, you may find the world of Apple a bit confusing at first. If this is the case, perhaps the best advice would be to spend some time with your Apple manuals. (Owners of the Apple //e may be interested in reading the two-volume set of manuals on Applesoft BASIC available through dealers.) While this short summary is no substitute for the manuals, it should be enough to get you started on *Nibble*'s program listings.

A QUICK OVERVIEW OF THE APPLE

When you first switch on your Apple, with a disk in the drive, you will most likely see a ']' character, called a prompt. The ']' prompt tells you that you can do one of three things:

1. Give commands in the disk command language (e.g. CATALOG).
2. Give commands in Apple's version of the BASIC language, Applesoft BASIC (e.g. PRINT 36+42).
3. Type in Applesoft BASIC program lines (e.g. 10 INPUT K).

To type in programs from *Nibble*, you may need to do all three.

ENTERING AN APPLESOFT BASIC PROGRAM

Before entering a program listing from *Nibble*, you should first thoroughly read the article accompanying the program. You may not understand all of the explanations the first time through, but be on the lookout for any special directions for typing the program. You should also be sure to have an initialized disk ready so that you can save your work. (See your disk drive manual or //e owner's manual for details on initializing disks.)

All BASIC programs consist of a sequence of program lines. Each program line begins with a number and is followed by one or more program statements separated by colons. For example:

```
20 FOR J = 1 TO 5:PRINT CHR$(7):
   NEXT J
```

To enter a program, begin with the first numbered line and type it in exactly as it appears (including the line number itself). Though the program line may span several printed lines in the listing, do not press <RETURN> (the RETURN key) until you reach the next line number. Then begin the process again with the next line number. When you reach the end of the program, save your work on the disk by typing the command SAVE followed by the name of the program. That's all there is to it!

LISTING 1

```
10 REM RING THE BELL
20 FOR J = 1 TO 5:PRINT CHR$(7)
    :NEXT J
30 END
```

To enter the sample program BELLS shown in **Listing 1**, you should follow this sequence:

1. Type the word NEW <RETURN> to clear memory of any old programs. (Make sure the <CAPS LOCK> key is down if you are using an Apple //e.)
2. Type line 10 exactly as it appears, but do not press <RETURN> until you have typed the last word in the line, "BELL".
3. Repeat this procedure with lines 20-30.
4. With an initialized disk in the drive, type SAVE BELLS <RETURN> to save your program on the disk.
5. Since the program is now in memory, you may just type RUN <RETURN> to start it. If you erase it from memory by running a different program or by turning off your computer, you may put it back into memory and start it again by typing the command RUN BELLS <RETURN>.

A FEW TIPS

The following tips may make your work a little easier:

1. If you make a mistake while typing, use the back arrow key to go back and correct it, and the forward arrow key to "retype" the remainder of the line before pressing <RETURN>. If you have already pressed <RETURN> before you catch your error, simply retype the entire line (number and all) and the new version will take the place of the old. (The use of an Applesoft line editor like MicroSPARC's GALE can eliminate much of this work.)
2. Be particularly careful when typing in statements that contain DATA. Typos in other lines will probably show up as SYNTAX ERRORS when the program is finally run, but those in DATA statements may not.
3. Save the program to disk periodically as you go along to minimize the effect of an accidental power loss.
4. Don't try to make your own modifications to the program until you have typed it in as published and have run it successfully. This will make it easier to debug in case you have made typing errors. MicroSPARC's KEY PERFECT program and the tables published along with program listings can be used to check your typing and report any lines containing errors. See the advertisement in this issue for details.
5. Lower-case letters may only be used within statements that begin with either REM or DATA, or between quotes.
6. If the program does not seem to run correctly, it may be helpful to temporarily remove any ONERR statements. This will allow you to see error messages suppressed by ONERR.
7. If you're certain that you have typed the program correctly, but you still can't get it working, call *Nibble*'s Technical Support Office at (617) 371-1660 for assistance.

ENTERING MACHINE LANGUAGE PROGRAMS

Both BASIC and the disk command language are powerful languages that interpret English-like words. Your Apple can also understand a much lower-level language, called machine language. Since this is the Apple's "native tongue," machine language programs perform much more quickly than those written in BASIC.

Often, a program called an assembler is used to help create machine language programs. An assembler first allows the programmer to write an assembly language program and then translates this program into machine language before it is run. Though you may not have an assembler, you will still be able to extract and use the machine language from *Nibble*'s listings. The advantage of an assembler is that it allows you to easily modify the program, or to "borrow" a programming technique.

If you don't own an assembler, you will need to enter machine language programs directly into the Apple's memory through what is called the System Monitor (not to be confused with your video monitor). To reach this level from the disk/BASIC level (indicated by the ']' prompt), you simply type CALL-151 and press <RETURN>. You will then see an asterisk (*), which is the prompt for the System Monitor. While you can use many commands at this level, the only one you will need to enter *Nibble* listings looks like this example:

```
300:A2 05 20 DD FB CA F0 03 4C 02
03 60 <RETURN>
```

In this command, the "300" specifies a memory location in your Apple and the colon tells the Apple to put the following number (A2, a number in base 16) into that location. The numbers following the first (05 through 60) are put into subsequent memory locations. Though you don't need to understand base 16 (or hexadecimal) numbers, you should know that all machine language numbers are given in hexadecimal notation.

A SAMPLE MACHINE LANGUAGE PROGRAM

Let's follow a short example of entering a machine language program. **Listing 2** shows the contents of a portion of the Apple's memory, often called a "hex dump." The number to the left of the hyphen is a memory location's "address," and the numbers to the right are the contents of that and subsequent memory locations. Often such a listing will be preceded by two numbers separated by a period (.). These numbers are the starting and ending addresses of the part of memory shown.

LISTING 2

```
 300.30B
0300- A2 05 20 DD FB CA F0 03
0308- 4C 02 03 60
```

Listing 3 shows the assembly language which was used to create the machine language program shown. Notice that the numbers in the left-hand columns look very similar to those in **Listing 2**. They are, in fact, the same set of memory addresses and their program contents in a different format. All of the columns on the right are assembly language instructions and comments. While other assemblers use slightly different formats, you will always be able to find the two columns which contain the addresses and contents of memory.

LISTING 3

```
                1 *RINGER PROGRAM
                2       ORG  $300
                3 BELL  EQU  $FBDD
0300: A2 05     4       LDX  #$5
0302: 20 DD FB 5 LOOP   JSR  BELL
0305: CA        6       DEX
0306: F0 03     7       BEQ  END
0308: 4C 02 03 8        JMP  LOOP
030B: 60        9 END   RTS
```

To enter the machine language listings, you just type in the addresses and their contents as follows:

1. Type CALL−151 <RETURN> to get into the System Monitor. You should now have an asterisk (*) prompt.
2. Type the first memory address shown, a colon (instead of the hyphen shown in the listing), and the memory contents. If you were using a listing similar to **Listing 2**, you would type:

   ```
   300:A2 05 20 DD FB CA F0 03
   <RETURN>
   308:4C 02 03 60 <RETURN>
   ```

   If you were using an assembler listing like that in **Listing 3**, you would type:

   ```
   300:A2 05 <RETURN>
   302:20 DD FB <RETURN>
   305:CA <RETURN>
   306:F0 03 <RETURN>
   308:4C 02 03 <RETURN>
   30B:60 <RETURN>
   ```

   Be sure that you do not put a space between the colon and the first pair of hexadecimal digits, but that you do put spaces between subsequent pairs. You may actually type up to 85 pairs of digits after each colon, but it is easier to follow the listing as published for your first time through.
3. When you have entered the entire listing, press <CTRL>C<RETURN> to get back to the disk/BASIC level indicated by the ']' prompt. This is accomplished by pressing the C key while holding down the Control key, and then pressing the <RETURN> key.
4. Although BASIC programs always start in the same place in the Apple's memory, and thus can simply be SAVEd, machine language programs can start at various places in memory. For this reason, the command to save a machine language program (BSAVE) must include the starting address (A) and the length (L) of the program being saved. For the program above, the command:

   BSAVE RINGER,A$300,L$C

   would be used. (The dollar sign ($) signifies that the number is given in hexadecimal notation.)

You can now run this program by typing BRUN RINGER. (The address and length are only necessary for the BSAVE command.) You can also run this program from the disk/BASIC level (after you have BLOADed it into memory) with a CALL statement followed by the decimal equivalent of the starting address. In this case, CALL 768 can be used to run the program since 768 is the decimal form of the number $300.

Sometimes a machine language listing is not a program at all, but is merely a table of data (such as a Hi-Res graphics shape table). In these cases, the memory addresses and their contents should be typed in as described above, but you should not attempt to BRUN the file you have saved. You will be able to determine whether the machine language listing is a program or a data table by reading the accompanying article.

As with Applesoft programs, MicroSPARC's KEY PERFECT program can help you find typing errors in machine language programs. Also, the *Nibble* program M.L.E. (Machine Language Editor) can simplify the entry and editing of machine language programs. (See the *Nibble* software catalog in the back of this issue for details.)

We hope that *Nibble* will provide you with useful, interesting software, and that your efforts to type in and customize the program listings will help you become more familiar with your Apple. Please feel free to let us know what's working for you, and how you think we could be even more helpful. We'd love to hear from you!

DISK OPERATING SYSTEMS

A disk operating system is a powerful program that allows a computer to communicate with devices which are hooked up to it, such as disk drives, printers, and even the keyboard and video display. There are two different disk operating systems for the Apple II, //e and //c: DOS 3.3 and ProDOS. While DOS 3.3 will run on any Apple II, ProDOS will only work on an Apple II Plus that has an extra 16K of RAM or an Apple //e or //c.

Although there are many similarities between the two operating systems, there are enough differences that some programs which work properly under one system, will not work under the other. Whenever possible, the programs published in *Nibble* will be written in such a way that they will work properly under both DOS 3.3 and ProDOS. You will be able to tell if this is the case by reading the section that contains author information on the first page of each article.

For more information about DOS 3.3 and ProDOS, check with your local Apple dealer or user's group.

In this issue, good programs for beginners appear in the One-Liners section on page 160.

TURTLE BASIC

by J.B. Ward

FEATURE ARTICLE

Add Logo-like commands to Applesoft for fast, easy turtle graphics programming. Turtle BASIC adds 24 keywords that let you control the "turtle," print Hi-Res text, and create animation.

"Once" said the Mock Turtle at last, with a deep sigh, "I was a real Turtle."* If your idea of a turtle is a short, slow, green reptile in a shell, you have not discovered the mock turtle of the modern age — turtle graphics. Turtle graphics is a concept that was developed as part of the Logo computer language. Logo's inventors felt that understanding of graphics programming could be improved if students could visualize moving an object around the screen, commanding it to move a certain distance or face a particular direction. To reinforce the point, they devised a simple robot (which looked a bit like a turtle) that could be programmed to move in the same way as the graphics cursor.

Turtle BASIC (**Listings 1-4**) modifies Applesoft to implement turtle graphics commands. Special keywords, such as FWD, BACK, LEFT and RIGHT, direct the turtle (or Hi-Res cursor) around the screen, while other keywords allow you to easily display Hi-Res text and create page-flip animation. Turtle BASIC is fast, and it eliminates the need for most shape tables and HPLOT statements.

Alice in Wonderland by Lewis Carroll.

J.B. Ward, Saint John, N.B., Canada. *Turtle BASIC requires 64K and works only under DOS 3.3.*

WHY A TURTLE?

There are two ways to draw a line on the Apple. One is to specify two sets of x,y coordinates and draw a line that connects them. This is called the

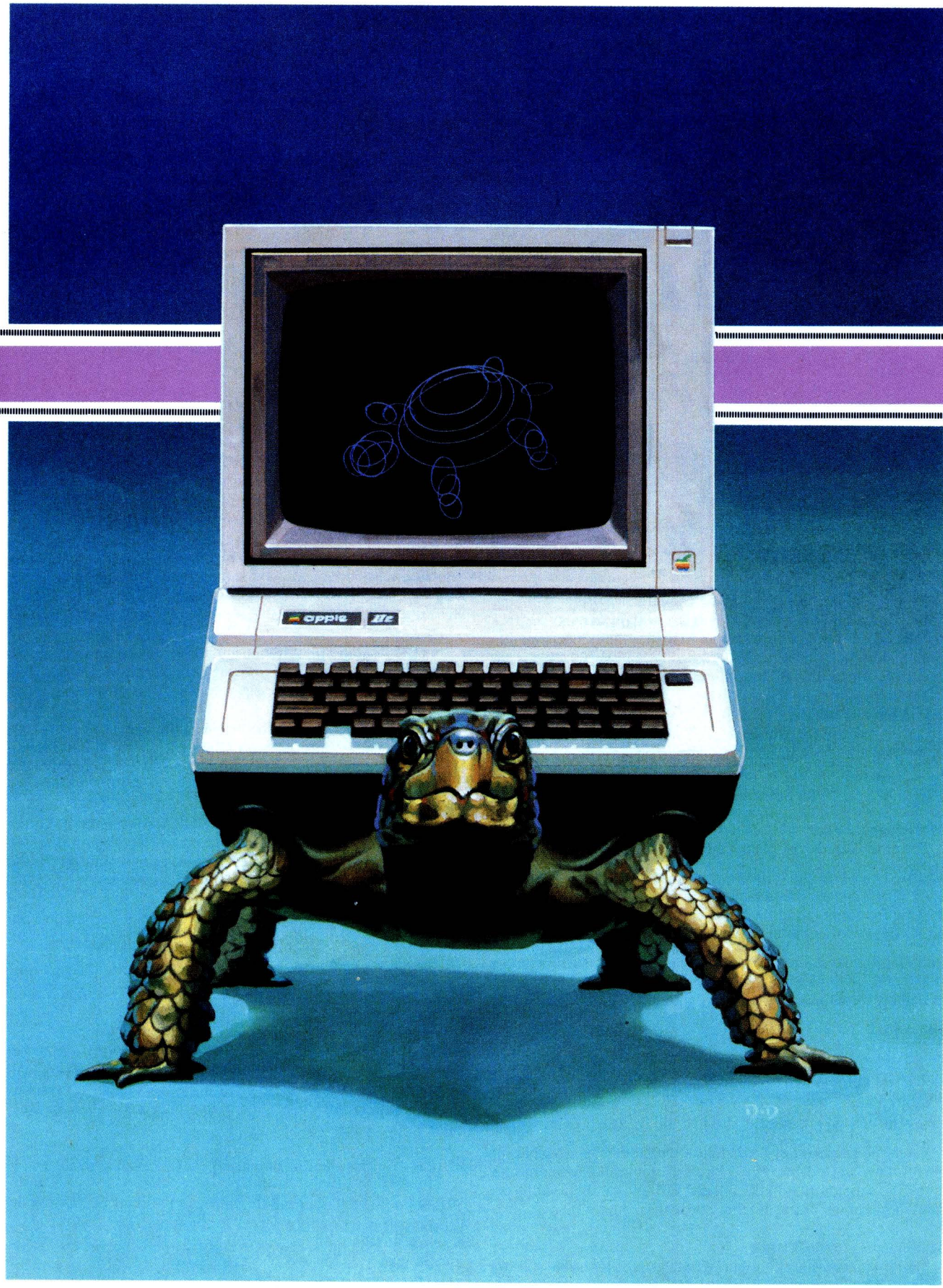

apple IIe

TABLE 1: List of Turtle BASIC Commands

Command	Description
FWD	Moves the turtle forward
BACK	Moves the turtle back
LEFT	Rotates the turtle's heading to the left
RIGHT	Rotates the turtle's heading to the right
JUMP	Moves the turtle to a specified position or series of positions
NEST	The upper-left corner of the screen
CENTER	The center of the screen (135,96)
TURNTO	Sets the turtle's angular heading
SHOW	Makes the turtle draw
HIDE	Makes the turtle stop drawing
HIRES	Selects Hi-Res graphics mode
TEXT	Selects text mode
SPLIT	Selects split graphics/text mode
FULL	Selects full-screen graphics
VIEW#	Selects Hi-Res page to view
PLT#	Selects Hi-Res page on which to plot
BLACK	Changes turtle color to black
WHITE	Changes turtle color to white
CLR	Clears the screen
DO	Subroutine call that allows a name
!	A REMark statement
TURTLEX	Current turtle x-coordinate (read only)
TURTLEY	Current turtle y-coordinate (read only)
ANGLE	Current turtle heading (read only)

Cartesian method, named for Rene Descartes. Another way (the turtle graphics way) is to say, "Turn left or right so many degrees and then go forward so many steps." Often, graphics that are difficult to draw using the Cartesian method become very simple using the turtle. Turtle BASIC lets you choose either method.

TALKING TURTLE

Before you can use the turtle, you need to learn its language. A complete list of the Turtle BASIC command set is shown in **Table 1**. The section PLAYING TURTLE leads you through a short sample session of programming in Turtle BASIC.

Moving the Turtle

The most basic keywords tell the turtle how to move. They are:

1. FWD *n* moves the turtle forward *n* units.
2. BACK *n* moves the turtle back *n* units.
3. LEFT *d* turns the turtle left *d* degrees.
4. RIGHT *d* turns the turtle right *d* degrees.

In these commands the *n* is specified in steps. The Hi-Res screen is 280 of these units wide and 192 high. The *d* represents an angle in degrees. Both *n* and *d* may be expressions. For instance:

```
10Ø FWD 5 : LEFT 9Ø
```

moves the turtle forward five units, and then turns it 90 degrees counterclockwise. But the following statements are also handled properly:

```
1ØØ FWD A + 5 * SQR(Z1) : LEFT DQ * 2.3
```

The turtle graphics keywords follow the same rules as Applesoft BASIC keywords. Turtle BASIC uses the same Hi-Res coordinate system as Applesoft, so HPLOT and DRAW can be used in the same program as Turtle BASIC commands.

Making the Turtle Visible

SHOW makes the turtle visible. When the turtle is visible, it draws on the screen as you move it around. HIDE makes the turtle invisible. When the turtle is invisible, no lines are drawn when you move it around.

Orienting the Turtle

The JUMP command moves the turtle to a particular position or series of positions. Each position is specified by its x,y coordinates. A line may or may not be drawn, depending upon the pen color, SHOW/HIDE status and the exact syntax used. JUMP is similar in meaning to Applesoft's HPLOT. Multiple arguments are possible, with each set of coordinates separated by the keyword TO. The following are examples of JUMP statements:

JUMP *x,y* — Positions the turtle at coordinates *x,y*. Using this syntax does not draw a line from the previous position to *x,y*, regardless of the SHOW/HIDE status. This takes care of the common case where a figure must be drawn, but the turtle is not at the starting point.

JUMP TO *x,y* — Draws a line between the previous position and *x,y*, if the turtle is in SHOW mode.

JUMP *x1,y1* TO *x2,y2* TO *x3,y3* — Draws a line from *x1,y1* to *x2,y2*, then to *x3,y3*.

JUMP TO *x1,y1* TO CENTER TO *x2,y2* TO NEST — Draws a line from the previous position to *x1,y1*, then to the center of the screen, then to *x2,y2*, then to the screen origin. Note that CENTER and NEST are keywords (discussed below).

TURNTO *d* — Turns the turtle to a heading of *d* degrees. Zero degrees points horizontally to the right. A positive angle is clockwise. The heading used here is an absolute heading, while the headings used by RIGHT and LEFT are relative to the current heading.

CENTER — Puts the turtle at the center of the screen (x,y = 140,96). A line is not drawn from the previous position. CENTER may be used by itself or as part of a JUMP command.

NEST — Puts the turtle at the origin in the upper-left corner of the screen (x,y = 0,0). As with CENTER, a line is not drawn. NEST may be used by itself or as part of a JUMP commannd.

Hi-Res Screen Control

The Applesoft keywords HGR and HGR2 are replaced in Turtle BASIC by more flexible keywords that control the Apple video soft switches directly from BASIC. Each switch can be independently manipulated so that page-flip animation is made easy.

HIRES — Selects Hi-Res graphics mode. HIRES has no effect on the split/full mode setting or the screen selected for plotting and viewing (see below).

TEXT — Selects text mode, page 1. Switching to text mode does not disturb plotting to either Hi-Res page.

SPLIT — Selects the split (or mixed text and graphics) Hi-Res mode on the screen currently selected for viewing. This allows the bottom four lines of the text screen to be displayed on the Hi-Res screen. SPLIT is very useful when you are experimenting with immediate mode turtle commands.

FULL — Selects the full (unmixed) Hi-Res display mode.

VIEW# *n* — Selects a page for viewing. This may or may not be the same page currently selected for plotting. If *n* is odd, page 1 is selected; otherwise, page 2 is selected. The value of *n* must be greater than −32769 and less than 32768.

PLT# *n* — Selects a Hi-Res page for plotting. If *n* is odd, page 1 is selected; if *n* is even, page 2 is selected.

CLR — Clears the plotting page and fills it with the opposite of the currently selected turtle color (either black or white).

BLACK and WHITE — Specify the color of the turtle. WHITE is the default color, and will cause the turtle to draw white lines on a black background. BLACK has the opposite effect. Note that turtle graphics are monochromatic: if a bit on the screen is on, it is considered white, regardless of how it may appear on a color monitor.

Hi-Res Printing

HPRINT is simply a high resolution version of the Applesoft PRINT. The only difference in syntax is that HPRINT accepts a set of x,y coordinates as input arguments. These coordinates specify the position of the upper-left corner of the first character to be printed. In the following examples, *list* is any set of string or numeric expressions that would be acceptable to Applesoft's PRINT. Upper-case and lower-case characters are included in the Hi-Res character set. As usual, *x* and *y* represent a set of coordinates.

HPRINT (*x,y*) *list* — Prints a *list* starting at *x,y*. A carriage return is performed at the end of the print.

HPRINT *list* — Prints a *list* at the current text position on the screen. If this is followed by an HPRINT (*x,y*) *list* statement, the first character in the *list* would be at coordinates 0, *y*+8, that is, at the beginning of the next line.

HPRINT (*x,y*) *list* ; — As with PRINT, a semicolon eliminates the carriage return at the end of output. If an HPRINT *list* statement followed this one, the second output string would begin immediately after the end of the first.

HPRINT (*x,y*) ; — Sets up a starting position for some future HPRINT.

HPRINT — Performs a carriage return.

HPRINT ; — Does nothing.

System Variables

Three special keywords — TURTLEX, TURTLEY and ANGLE — provide information about the status of the turtle. They act like variables that cannot be assigned a value directly. Unlike Applesoft variable names, where only the first two characters are significant, all characters of these names are significant. TURTLEX and TURTLEY supply information about the position of the turtle, while ANGLE supplies the heading. For instance, to get the current turtle heading, you could just type:

```
PRINT ANGLE
```

or you could assign the value to a variable:

```
TH = ANGLE
```

but you cannot assign a value to ANGLE or expect to change the current heading with a statement such as:

```
ANGLE = 45
```

In this case, the BASIC variable AN gets the value 45 and the turtle heading does not change. CURSORX and CURSORY can be used in the same way as ANGLE.

Miscellaneous Keywords

DO *name* — Is an extra type of subroutine call that is similar to GOSUB but accepts variable names. The variable must contain the line number of the subroutine being called.

This new statement improves program readability. GOSUB SQUARE beats GOSUB 2000 for readability, but a simple DO SQUARE is even better. When using DO, try to keep all variable definitions together, near the beginning of the program. This helps you keep track of your subroutine locations, and it speeds program execution, since the BASIC interpreter has fewer program lines to search through to find the value.

Exclamation Mark (!) — Substitutes for the REM statement. Like DO, it helps make programs more readable. The old REM has been left intact, and may still be used.

TABLE 2: Character Table for the Letter E

Table Offset (dec, hex)	Hex Table Address	Bit Pattern	Hex Value	Dec Value
296, $128	$92F0	00111110	$3E	62
297, $129	$92F1	00000010	$02	2
298, $12A	$92F2	00000010	$02	2
299, $12B	$92F3	00011110	$1E	30
300, $12C	$92F4	00000010	$02	2
301, $12D	$92F5	00000010	$02	2
302, $12E	$92F6	00111110	$3E	62
303, $12F	$92F7	00000000	$00	0
		↑ (always 0)		

Custom Character Sets

Since the character set used by HPRINT is kept in RAM, it is possible to change individual characters in a set, or substitute another character set of your own. The character table is an array 96 characters long, with eight bytes per character. The address of the beginning of this table is at location $8A8B (decimal 35467). Using an instruction sequence such as:

```
1000 PRINT CHR$(4);"BLOAD CSET2,A24576"
1010 POKE 35467,24576-INT(24576/256)*256
1020 POKE 35468,24576/256
```

would allow you to switch to a new character set kept at $6000 (24576). After that, you could switch back and forth using only two POKE instructions.

To go back to the default character set, use:

```
POKE 35467,200 : ! Low byte address,$C8
POKE 35468,145 : ! High byte address,$91
```

Because the starting address of the default set is $91C8 (decimal 37320), breaking $91C8 into two bytes gives $C8 (decimal 200) and $91 (decimal 145).

The table starts with the space character, $20, and ends with the rubout character (a solid block), $7F. Since each character is represented by eight bytes, the first eight bytes define a space, the next eight define an exclamation mark (!), and so on.

The offset into the table for the letter E is given by:

OFFSET = 8*(ASC("E")−32)

which equals decimal 296.

Table 2 gives the bit pattern for the letter E. Wherever a bit is set to one, a pixel on the Hi-Res screen is lit. You can see that the bit patterns must be backwards. Less obvious is that while the bit patterns are all eight bits wide, a character on the screen is only seven bits wide. Bit 7 of each byte is never used, and should be left clear.

Your new character set can be made of graphic symbols or alphabetic characters. Large, complex symbols can then be made by put-

ting a number of characters together; each character is a piece of the larger symbol.

PLAYING TURTLE

Perhaps the best way to get the feel of turtle graphics programming is to try it in immediate mode, so you can see the effect on the screen as you enter each command. Start by initializing Turtle BASIC with:

```
BRUN TURTLE
NEW
```

Then enter a very short program:

```
10 FWD 50 : LEFT 90
20 END
```

Set up the Hi-Res, split screen for both viewing and plotting; clear it; and move the turtle to the center of the screen as follows:

```
VIEW# 1
PLT# 1
SPLIT : HIRES
CLR
CENTER : TURNTO 0
```

Now you can draw a square, one side at a time, by typing:

```
RUN
RUN
RUN
RUN
```

By changing the heading, you can draw a new square tilted at 45 degrees:

```
LEFT 45
RUN
RUN
RUN
RUN
```

Finally, you can label your creation with:

```
HPRINT(10,50) "SQUARES"
```

The two-line program simply draws a side of the square and turns the turtle 90 degrees to the left. See the demonstration programs (**Listings 5, 6 and 7**) for examples of incorporating these and other commands into a program. DEMO 1 uses a subroutine that draws a square in a manner very similar to what you just did in immediate mode. (See **Figure 1** for the resulting display.) DEMO2 demonstrates the use of subroutines to create complex figures on both graphics pages, and then uses the VIEW command to achieve page-flip animation (**Figure 2**). Notice the use of Hi-Res text to include a prompt on the Hi-Res screen. DEMO3 uses simple turtle graphics commands to create an abstract pattern.

ENTERING THE PROGRAMS

Turtle BASIC is listed in two different ways. The first is as source code for entry using MicroSPARC's Assembler (**Listings 1, 2 and 3**), and the second is as a hex dump of the assembled program (**Listing 4**). If you don't have the MicroSPARC Assembler, follow the directions for entering the program from the hex dump.

FIGURE 1: Squiral Created by DEMO1

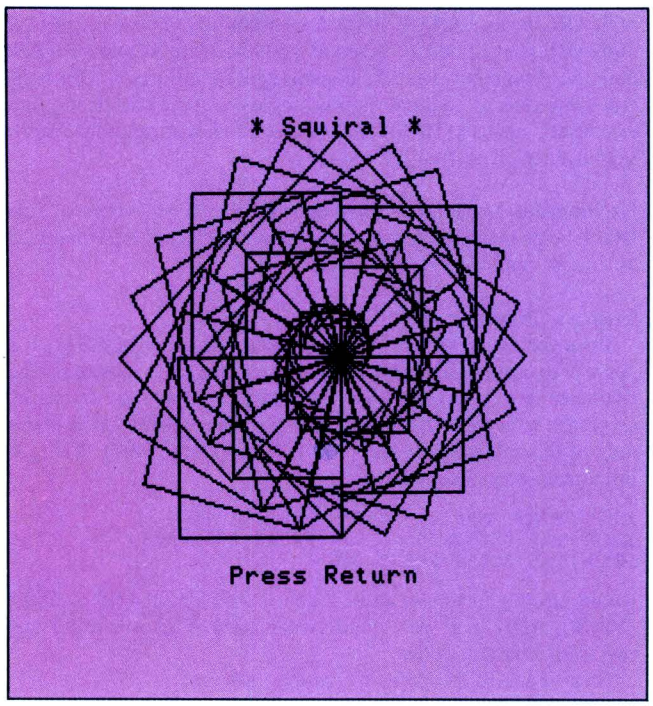
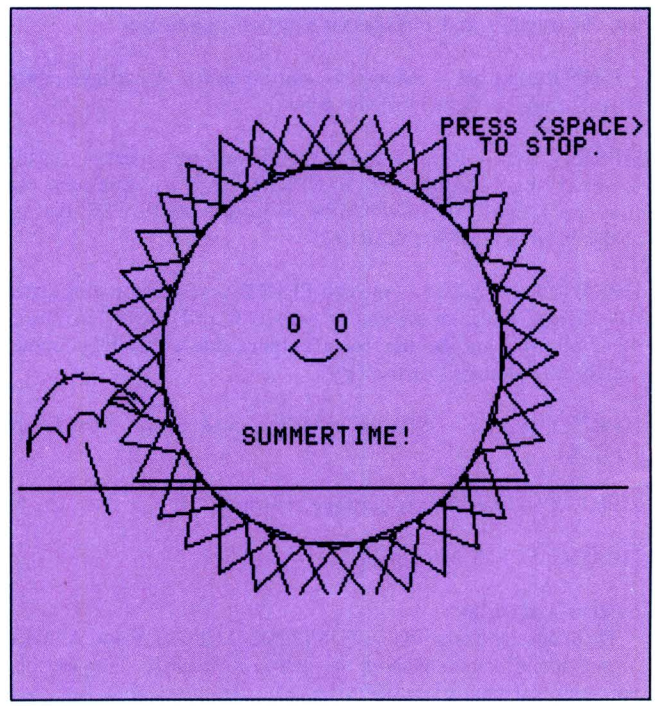


FIGURE 2: Animation Display Created by DEMO2



Entering the Program With the MicroSPARC Assembler

If you are using a one-drive system, use the FID program from your DOS 3.3 System Master disk to copy the file MACLIB .SOURCE from the MicroSPARC Assembler disk to the disk that will hold your other source files.

Using the Assembler's editor, enter **Listing 1** and save it under the name TURTLE1. Next, enter **Listing 2** and save it under the name TURTLE2. Finally, enter **Listing 3** and save it under the name TURTMAC. These files will show up in the disk directory with a .SOURCE suffix, but you should not enter the suffix when saving or loading these files with the Assembler. Load TURTLE1 and assemble it. It will automatically call the other two source files. When the assembly is completed, the object file, TURTLE, will have been automatically saved on the disk.

Entering the Program From the Hex Dump

To key in Turtle BASIC without the MicroSPARC Assembler, enter the Monitor with CALL −151 and type the hex code as shown in **Listing 4**. Save the program with the command:

BSAVE TURTLE,A$8800,L$CC8

Entering the Demonstration Programs

Because Turtle BASIC is an altered version of Applesoft, it should be initialized with the command BRUN TURTLE before any Turtle BASIC program is entered (or loaded from disk). Once this has been done, enter DEMO1 (**Listing 5**) and save it with the command:

SAVE DEMO1

Enter DEMO2 (**Listing 6**) and save it with the command:

SAVE DEMO2

Enter DEMO3 (**Listing 7**) and save it with the command:

SAVE DEMO3

For help in entering *Nibble* listings, see "A Welcome to New *Nibble* Readers" at the beginning of this issue.

Turtle BASIC, Nibble Calculator and Tank Combat are available on diskette for an introductory price of $19.95 plus $1.50 shipping/handling ($2.50 outside the U.S.) from Nibble, 45 Winthrop St., Concord, MA 01742. Introductory price expires 3/31/86.

LISTING 1: TURTLE1 — Source File for MicroSPARC Assembler

```
 0 ;
 1 ;**********************
 2 ;*      TURTLE1       *
 3 ;*    TURTLE.BASIC    *
 4 ;*    BY J.B. WARD    *
 5 ;* COPYRIGHT (C) 1986 *
 6 ;* BY MICROSPARC, INC *
 7 ;* CONCORD, MA  01742 *
 8 ;**********************
 9 ;*
10 ;* MICROSPARC ASSEMBLER
11 ;*
12          USE   MACLIB,D1
13          USE   TURTMAC,D2
14          UEN
15          MUL
16 ;  Define BASIC entry points &
17 ;  page zero locations.
18 NEW      EQU   $D64B     ; "New-workspace" routine
19 WARM     EQU   $E003     ; Warm-start entry
20 IEVAL    EQU   $E6F8     ; Evaluate-exp., return 1 byte.
21 FEVAL    EQU   $DD67     ;  "          ", "    " flt.pt.
22 SINE     EQU   $EFF1     ; sub: Sine
23 COSINE   EQU   $EFEA     ; sub: Cosine
24 FLOAD    EQU   $EAF9     ; Sub: load FACC1
25 FSAVE    EQU   $EB2B     ; Sub: save FACC1
26 FADD     EQU   $E7BE     ; sub: flt.pt. add
27 FADHAF   EQU   $E7A0     ; sub: add 0.5 to FACC1
28 FSUB     EQU   $E7A7     ; sub: flt.pt. subtraction
29 FIX      EQU   $E10C     ; sub: convert FACC1,fp-to-int.
30 FMPY     EQU   $E97F     ; sub: flt.pt. multiply
31 FDIV     EQU   $EA66     ; sub: flt.pt. divide
32 INT      EQU   $EC23     ; sub: take int. part of FACC1
33 REM      EQU   $D9DC     ; sub:  "REM" action
34 GOTO1    EQU   $D941     ; sub:  "GOTO" action
35 CHKDEP   EQU   $D3D6     ; ???
36 NEXT     EQU   $D7D2     ; "perform-next-line" entry
37 DOSWRM   EQU   $9DD0     ; DOS3.3 warm-start entry
38 CHKRPA   EQU   $DEB8     ; sub: check & skip right-paren
39 CHKLPA   EQU   $DEBB     ; sub: check & skip left-paren
40 CHKCOM   EQU   $DEBE     ; sub: check & skip comma
41 CHKC     EQU   $DEC0     ; sub: check & skip chr.in Areg
42 CHKALF   EQU   $E07D     ; sub: if Areg=letter,carry:=1
43 PRINT2   EQU   $DAD5     ; sub: "PRINT" entry
44 ILLQTY   EQU   $F206     ; "illegal quantity" msg. entry
45 BELL     EQU   $FBDD     ; sub: ring bell.
```

continued on next page

LISTING 1: TURTLE1 *(continued)*

```
 46 GETC2    EQU   $00B1      ; sub: get-next-nonblk-input-chr
 47 GETC3    EQU   $00B7      ; sub: get-last-nonblk-input-chr
 48 HRPAGE   EQU   $E6        ; Hires-plot page;  $20 or $40
 49 LASCHR   EQU   $B8        ; &B9 : ptr. to last input char.
 50 LINENO   EQU   $75        ; BASIC executing line #.
 51 CSWL     EQU   $36        ; char.output vector,low
 52 CSWH     EQU   $37        ;  "     "     "    ,high
 53 GBASL    EQU   $26        ; hires graphics base addr,low
 54 GBASH    EQU   $27        ;  "       "      "   "  ,high
 55 NTOKES   EQU   19         ; Number of new token types.
 56 TOTOKE   EQU   193        ; "TO" token number
 57 CENTOKE  EQU   $F2        ; "CENTER" token number
 58 NESTOKE  EQU   $F3        ; "NEST" token number
 59 ;      temp. plotting vars; "borrows" F.P.registers.
 60 PMSK     EQU   $9D        ;"plot-mask";dot to be plotted.
 61 XOFSET   EQU   $9E        ;x byte-offset into hires screen
 62 PTCTR    EQU   $A0        ;point counter;# points to plot.
 63 XSTRT    EQU   $A2        ; starting x byte offset
 64 MSKIX    EQU   $A3        ; index into dot-mask table
 65 ;  $A4 must not be touched,else sys goes strange.
 66 TMPIXC   EQU   $A5        ; temp.integer x-coord
 67 TMPIYC   EQU   $A7        ; temp.integer y-coord
 68 INCVAL   EQU   $A9        ;(When: TSTSUM := TSTSUM+INCVAL)
 69 TSTSUM   EQU   $AB        ;(results in carry-set,then we)
 70 ;              (change dependent coord by +-1.)
 71 ;              (Else next pt.is on vert.or horiz.)
 72 ;              (line from last point plotted.)
 73          ORG   $8800
 74 ;  * Begin: Patch BASIC, initialize, warm-start BASIC.
 75 TURTLE   JSR   CPYROM     ;Copy rom-to-ram
 76          JSR   PATCH      ;Patch ram copy to enter turtle.
 77          JSR   NEW        ;Call BASIC "new" routine.
 78          JSR   SETMEM     ;Set himem & lomem.
 79          JSR   SETRAM     ;Set flags & capture reset vec.
 80          JMP   WARM       ;Warm-start interpreter.
 81 ;  Function jump table
 82 ;             Function   Token #
 83 FNADRS   ADDR  FWD-1      ;EC
 84          ADDR  BACK-1     ;ED
 85          ADDR  LEFT-1     ;EE
 86          ADDR  RIGHT-1    ;EF
 87          ADDR  SHOW-1     ;F0
 88          ADDR  HIDE-1     ;F1
 89          ADDR  CENTER-1   ;F2
 90          ADDR  NEST-1     ;F3
 91          ADDR  HEAD-1     ;F4
 92          ADDR  JUMPTO-1   ;F5
 93          ADDR  SPLIT-1    ;F6
 94          ADDR  FULL-1     ;F7
 95          ADDR  HIRES-1    ;F8
 96          ADDR  VIEW-1     ;F9
 97          ADDR  EXCLAIM-1  ;FA
 98          ADDR  DO-1       ;FB
 99          ADDR  HPRINT-1   ;FC
100          ADDR  BLACK-1    ;FD
101          ADDR  WHITE-1    ;FE
102 ;  Token names for turtle extensions
103 NAMES    DCI   FWD"
104          DCI   BACK"
105          DCI   LEFT"
106          DCI   RIGHT"
107          DCI   SHOW"
108          DCI   HIDE"
109          DCI   CENTER"
110          DCI   NEST"
111          DCI   TURNTO"
112          DCI   JUMP"
113          DCI   SPLIT"
114          DCI   FULL"
115          DCI   HIRES"
116          DCI   VIEW#"
117          DCI   !"
118          DCI   DO"
119          DCI   HPRINT"
120          DCI   BLACK"
121          DCI   WHITE"
122          DFC   0
123 ;  Copy all of $D000-$FFFF from ROM to RAM.
124 CPYROM   LDY   #0
125          STY   $50
126          LDA   #$D0
127          STA   $51
128          LDA   $C081      ;enable read-rom/write-ram
129 CPY010   LDA   ($50),Y
130          STA   ($50),Y
131          INY
132          BNE   CPY010
133          INC   $51
134          BNE   CPY010
135          RTS
136 ;  Patch RAM copy of interpreter to
137 ;  cause entry to turtle code.
138 PATCH    LDA   #$4C       ;Put JMP opcode at:
139          STA   $D60B      ;1) Parse patch
140          STA   $D737      ;2) List patch
141          STA   $D846      ;3) Run patch
142          STA   $D5E5      ;4) Parser "!" patch
143          STA   $DED5      ;5) Variable-search patch
144          LDA   #PARPAT    ;Fill in target addresses
145          STA   $D60C
146          LDA   #PARPAT/
147          STA   $D60D
148          LDA   #LISPAT
149          STA   $D738
150          LDA   #LISPAT/
151          STA   $D739
152          LDA   #RUNPAT
153          STA   $D847
154          LDA   #RUNPAT/
155          STA   $D848
156          LDA   #REMPAT
157          STA   $D5E6
158          LDA   #REMPAT/
159          STA   $D5E7
160          LDA   #VARPAT
161          STA   $DED6
162          LDA   #VARPAT/
163          STA   $DED7
164          LDX   #6
165 PAT010   LDA   NNS,X      ;"CLR" and "DRAW"
166          STA   $D109,X
167          DEX
168          BPL   PAT010
169          LDA   #PLOT-1
170          STA   $D020
171          LDA   #PLOT-1/
172          STA   $D021
173          LDA   #CLR-1
174          STA   $D022
175          LDA   #CLR-1/
176          STA   $D023
177          LDA   #TEXT-1    ;Change addr of TEXT
178          STA   $D012
179          LDA   #TEXT-1/
180          STA   $D013
181          LDA   #$FF       ;allow lower-case input
182          STA   $FD83
183          LDY   #5         ;fill in ext.exec.vectors
184 PAT020   LDA   EXADRS,Y
185          STA   EXLIAD,Y
186          DEY
187          BPL   PAT020
188          LDA   $C080      ;Write-protect RAM above $D000
189          LDA   $C080
190          RTS
191 NNS      DCI   PLT#"
192          DCI   CLR"
193 ;  Set high memory bound for interpreter.
194 SETMEM   LDA   #TURTLE-1 ;Himem=just below turtle.
195          STA   $73
196          STA   $6F
197          LDA   #TURTLE-1/
198          STA   $74
199          STA   $70
200          RTS
201 SETRAM   LDA #$FF
202          STA   $E4        ;Hcolour=white(7)
203          LDA   #0
204          STA   SHOFLG     ;'Show' turtle
205          LDA   #$20
206          STA   HRPAGE     ;Hires screen=1
207          JSR   CENTER     ;Center turtle.
208          JSR   WHITE      ;Set ink = white
209          LDA   #CHRTAB
210          STA   CTABAD     ;Point hires print to
211          LDA   #CHRTAB/   ;our hires char.table
212          STA   CTABAD+1
213          LDA   #RESET
214          STA   $3F2
215          LDA   #RESET/    ;Capture reset vector
216          STA   $3F3
217          EOR   #$A5
218          STA   $3F4
219          RTS
220 ;    ****************
221 ;    ***  PATCHES  ***
222 ;    ****************
223 ;  Note "hooks" for future extensions to
224 ;  keywords by inclusion of external code.
225 EXADRS    ADDR  RETURN     ;default addrs. for exec.
226          ADDR  RETURN     ;vectors,below.
227          ADDR  RUN990
228 EXLIAD    DFS   2          ;ext.exec.vectors
229 EXPAAD    DFS   2
230 EXRUAD    DFS   2
231 EXLIST    JMP   (EXLIAD)
232 EXPARSE   JMP   (EXPAAD)
233 EXRUN     JMP   (EXRUAD)
234 RETURN    RTS              ;"dummy" statement
235 ;  Reset patch
236 RESET     LDA   $C080      ;Set lang.card ram for read,
237          LDA   $C080      ;but write protected; Warm
238          JMP   DOSWRM     ;start interpreter.
239 ;  Program-list patch
240 LISPAT    CMP   #$FF       ;double token?
241          BNE   LIS010     ;-skip if not; else exec.
```

```
242             JSR   EXLIST     ;external code, if any.
243             BCC   LIS030     ;-br.if external token.
244 LIS010      CMP   #235       ;is it one of our tokens?
245             BCS   LIS020     ;(branch if so.)
246             SBC   #$7E       ;no.exec patched-over code&ret.
247             TAX
248             JMP   $D73A
249 LIS020      SBC   #235       ;Token probably ours.Point to
250             TAX              ;our name table,return to check.
251             STY   $85
252             LDA   #NAMES
253             STA   $9D
254             LDA   #NAMES-$100/
255             STA   $9E
256 LIS030      JMP   $D744
257 ;  Parser patch
258 PARPAT      LDA   $0F        ;see if our keyword table
259             CMP   #233       ;has been checked yet.
260             BCC   PAR010     ;if not,branch; else exec.
261             JSR   EXPARSE    ;external code,if any.
262             BCC   PAR020     ;br.if found in ext.table.
263             LDA   $0200,X    ;-name not in our table either.
264             JMP   $D60E
265 PAR010      LDA   #NAMES-1   ;Have parser check
266             STA   $9D        ;our name table.
267             LDA   #NAMES-1/
268             STA   $9E
269 PAR020      LDY   #0
270             JMP   $D5A8
271 ;  Run-time patch
272 RUNPAT      CMP   #$FF       ; double token?
273             BNE   RUN010     ;skip if not; else execute
274             JMP   EXRUN      ;some external code.
275 RUN010      SEC
276             SBC   #108       ;Is token one of ours?
277             CMP   #NTOKES
278             BCS   RUN990     ;Nope...must be an error.
279             ASL
280             TAY              ;Cvt.new token code to fn.table
281             LDA   FNADRS+1,Y ;offset.addr will be popped and
282             PHA              ;used when "$00B1" routine does
283             LDA   FNADRS,Y    ;an RTS instruction.
284             PHA
285             JMP   $00B1
286 RUN990      JMP   $DEC9      ; - Syntax Error.
287 ;  Patch for REMark alias, "!".
288 ;  Patch parser to include check for "!"
289 ;  immediately after check for "REM" fails.
290 REMPAT      BEQ   REM010     ; ( if token = REM )
291             CMP   #72        ; Not REM; how about
292             BEQ   REM010     ; alias, "!" ?
293             JMP   $D56D      ; -Was neither. Act nonchalant.
294 REM010      LDA   #0         ;Bingo. Token was either
295             STA   $0E        ;REM or !. We could care
296             JMP   $D5E9      ;less which; do REM action.
297 ;  Patch to variable-name search routine
298 ;  to allow use of "TURTLEX","TURTLEY",
299 ;  and "ANGLE" as special system variables.
300 VARPAT      LDY   #6         ;check for "TURTLEX"
301 VP010       LDA   ($B8),Y
302             CMP   TXN,Y
303             BNE   VP020
304             DEY
305             BPL   VP010
306             LDX   #XC
307             LDY   #XC/
308             JMP   VP040
309 VP020       LDY   #6         ;check for "TURTLEY"
310 VP030       LDA   ($B8),Y
311             CMP   TYN,Y
312             BNE   VP050
313             DEY
314             BPL   VP030
315             LDX   #YC
316             LDY   #YC/
317 VP040       JSR   GETC2      ;skip over name.
318             BCC   VP040
319             JSR   CHKALF
320             BCS   VP040
321             LDA   #0
322             STA   $11        ;clear "string" flag
323             STA   $12        ;clear "integer" flag
324             TXA
325             JMP   $DED8
326 VP050       LDY   #4         ;check for "ANGLE"
327 VP060       LDA   ($B8),Y
328             CMP   ANN,Y
329             BNE   VP070
330             DEY
331             BPL   VP060
332             FDIV  THETA,PIBY180,FTMP
333             LDA   #FTMP
334             LDY   #FTMP/
335             JMP   VP040
336 VP070       JSR   $DFE3      ;not turtlex,y or angle.
337             JMP   $DED8      ;do normal search & return.
338 TXN         DFC   $54,$55,$52,$54,$4C,$45,$58
339 TYN         DFC   $54,$55,$52,$54,$4C,$45,$59
```

continued on next page

LISTING 1: TURTLE1 *(continued)*

```
340 ANN      DFC   $41,$4E,$47,$4C,$45
341 TWOPI    DFC  $83,$49,$0F,$DA,$A1  ;PI*2
342 PIBY180  DFC  $7B,$0E,$FA,$35,$12  ;PI/180
343 INK      DFS   1          ;Turtle "colour";black or white
344 SHOFLG   DFS   1          ;Turtle "show/hide" flag
345 IXC      DFS   2          ;Turtle x-coord., integer
346 IYC      DFS   2          ;Turtle y-coord., integer
347 XC       DFS   5          ;Turtle X-coordinate,flt.pt.
348 YC       DFS   5          ;Turtle y-coord., flt.pt.
349 THETA    DFS   5          ;Directional angle
350 CTABAD   DFS   2          ;Hires-print char.table address
351 HPRX     DFS   2          ;Hires-print X-coord
352 HPRY     DFS   1          ;"          " Y-coord
353 CSWSAV   DFS   2          ;CSW (char.out.vec.) temp.save
354 DIST     DFS   5          ;Flt.pt. dist.var.for FWD/BACK
355 FTMP     DFS   5          ;Misc. FP.& int. scratch vars
356 FTMP2    DFS   5
357 PINC     DFS   5          ;Plot-increment
358 DX       DFS   5          ;Plot delta-x
359 DY       DFS   5          ;Plot delta-y
360 XDIR     DFS   1          ;X-direction; + or -
361 YDIR     DFS   1          ;Y-direction; + or -
362 X1TMP    DFS   2
363 Y1TMP    DFS   2
364 X2TMP    DFS   2
365 Y2TMP    DFS   2
366 ; Get a character from source input line for
367 ; inspection, only. Do not increment pointer.
368 GETC1    LDY   #0
369          LDA   ($B8),Y
370          RTS
371 ; Wait for vertical sync. pulse.
372 ; Used to synchronize page swapping.
373 WAITVID  PHA
374 WVD010   LDA   $C019
375          BMI   WVD010
376          PLA
377          RTS
378 ;  Get & evaluate expression in line.
379 ;  Return a 16-bit value: A=lobyte, Y=hibyte.
380 EVAL     JSR   FEVAL
381          JSR   FIX
382          LDA   $A1
383          LDY   $A0
384          RTS
385 ;  Set "colour" of turtle - draw with
386 ;  black or white stroke.
387 ;  Syntax :  [ WHITE ]     [ BLACK ]
388 ;       Set ink = white
389 WHITE    LDA   #0
390          STA   INK
391          RTS
392 ;      Set ink = black
393 BLACK    LDA   #$FF
394          STA   INK
395          RTS
396 ;  Move turtle forward or backward.
397 ;  Syntax: [ FWD <exp> ]    [ BACK <exp> ]
398 ;  Where <exp> is any valid algebraic expression
399 BACK     FEVAL DIST
400          CHS   DIST
401          JMP   BANDF
402 FWD      FEVAL DIST
403 BANDF    FLOAD THETA
404          JSR   COSINE     ;tmp := cos(theta)
405          FMPY  DIST       ;tmp := d*cos(theta)
406          FSAVE DX         ;( save delta-x )
407          FADD  XC         ;tmp.xc := c+d*cos(theta)
408          FSAVE FTMP
409          FLOAD THETA
410          JSR   SINE
411          FMPY  DIST
412          FSAVE DY
413          FADD  YC
414          FSAVE FTMP2
415          JSR   DRAWLIN    ;* plot a line *
416          RTS
417 ;  Turn turtle left or right, or
418 ;  head in some direction.
419 ;  Syntax: [ LEFT <deg> ]  [ RIGHT <deg> ]
420 ;          [ TURNTO <deg> ]
421 ;  <deg> specifies angle in degrees.
422 HEAD     FEVAL THETA
423          FMPY  PIBY180
424          JMP   MOD2PI
425 LEFT     FEVAL FTMP
426          CHS   FTMP
427          FLOAD FTMP
428          JMP   LANDR
429 RIGHT    FEVAL FTMP
430 LANDR    FMPY  PIBY180
431          FADD  THETA
432 MOD2PI   FSAVE FTMP
433          FLOAD TWOPI
434          FDIV  FTMP
435          JSR   INT
436          FMPY  TWOPI
437          FSUB  FTMP
438          FSAVE THETA
439          RTS
440 ;  Show turtle
441 SHOW     LDA   SHOFLG
442          AND   #$7F
443          STA   SHOFLG
444          RTS
445 ;  Hide turtle
446 HIDE     LDA   SHOFLG
447          ORA   #$80
448          STA   SHOFLG
449          RTS
450 ;  For HIRES,TEXT,SPLIT,FULL:
451 ;  Syntax: [ name ]    ... No parameters
452 HIRES    JSR   WAITVID
453          LDA   $C050      ;graphics
454          LDA   $C057      ;hires
455          RTS
456 TEXT     JSR   WAITVID
457          LDA   $C051      ;text
458          LDA   $C054      ;page 1
459          RTS
460 SPLIT    JSR   WAITVID
461          LDA   $C053
462          RTS
463 FULL     JSR   WAITVID
464          LDA   $C052
465          RTS
466 ;  Select hires screen for display
467 ;  Syntax: [ VIEW# <exp> ] ... If <exp>
468 ;  is odd, select screen1; else screen2.
469 VIEW     JSR   EVAL
470          JSR   WAITVID
471          LSR
472          BCS   PAGE1
473          LDA   $C055      ;select page 2
474          RTS
475 PAGE1    LDA   $C054
476          RTS
477 ;  Select hires screen for plotting
478 ;  Syntax:  [ USE# <exp> ] ... if <exp>
479 ;  is odd, select screen1; else screen2.
480 PLOT     JSR   EVAL
481          LSR
482          BCS   PLOT1
483          LDA   #$40
484          STA   HRPAGE
485          RTS
486 PLOT1    LDA   #$20
487          STA   HRPAGE
488          RTS
489 ;  Clear the hires screen presently
490 ;  selected for plotting.
491 ;  Syntax: [ CLR ]
492 CLR      CLC
493          LDA   #$1F
494          TAX
495          ADC   HRPAGE
496          STA   GBASH
497          LDA   #0
498          LDY   #$FF
499          STA   GBASL
500          BIT   INK
501          BVC   CLR10
502          TYA
503 CLR10    STA   (GBASL),Y
504          DEY
505          BNE   CLR10
506          STA   (GBASL),Y
507          DEC   GBASH
508          DEX
509          BPL   CLR10
510          RTS
511 ;  Put turtle at centre of screen, or
512 ;  at origin
513 ;  Syntax:  [ CENTER ]    [ NEST ]
514 CENTER   LDX   #9
515 CEN010   LDA   CCORDS,X
516          STA   XC,X
517          DEX
518          BPL   CEN010
519          LDX   #3
520 CEN020   LDA   ICORDS,X
521          STA   IXC,X
522          DEX
523          BPL   CEN020
524          RTS
525 CCORDS   DFC   $88,$0C,$00,$00,$00 ; 140.0
526          DFC   $87,$40,$00,$00,$00 ; 96.0
527 ICORDS   ADDR  140
528          ADDR  96
529 NEST     LDX   #9
530          LDA   #0
531 NES010   STA   XC,X
532          DEX
533          BPL   NES010
534          LDX   #3
```

```
535 NES020   STA   IXC,X
536          DEX
537          BPL   NES020
538          RTS
539 ;  Alias for REMark statement
540 ;  Syntax:  [  ! <comments>  ]
541 EXCLAIM  JMP  REM
542 ;  Perform a BASIC subroutine call with
543 ;  an expression specifying the target
544 ;  line.  Syntax:  [ DO <exp> ]  Where
545 ;  <exp> is any valid expression resolvable
546 ;  to a 16-bit integer.
547 DO       LDA   #3
548          JSR   CHKDEP     ;Check nesting depth.
549          LDA   LASCHR+1   ;Push last-character ptr.
550          PHA
551          LDA   LASCHR
552          PHA
553          LDA   LINENO+1   ;Push current line number.
554          PHA
555          LDA   LINENO
556          PHA
557          LDA   #$B0       ; ??
558          PHA
559          JSR   EVAL       ;Get destination (target)
560          STA   $50        ;line number.
561          STY   $51
562          JSR   GOTO1      ;Do it !
563          JMP   NEXT
564 ;  Position turtle at absolute x,y coord
565 ;  Syntax:  [ JUMP  <Xexp> , <Yexp>  ]
566 ;    [ JUMP <Xexp> , <Yexp> TO <Xexp> , <Yexp> TO ... ]
567 ;   Example: the following expression is valid.
568 ;    JUMP TO CENTER TO NEST TO X1,Y1 TO SQR(A)+37,15.4
569 JUMPTO   LDX   #$FF
570          CMP   #TOTOKE    ;do we have a TO token ?
571          BNE   JMP05
572          INX
573 JMP05    STX   DIST       ;( just a temporary loc...)
574          BNE   JMP20      ;branch if not a TO token.
575 JMP10    JSR   GETC2      ;ignore TO, check next token/chr
576 JMP20    CMP   #CENTOKE   ;is it a CENTER token ?
577          BNE   JMP30
578          JSR   GETC2      ;( throw token away.)
579          JSR   JCENTR
580          JMP   JMP50
581 JMP30    CMP   #NESTOKE   ;is it a NEST token ?
582          BNE   JMP40
583          JSR   GETC2      ;( throw token away.)
584          JSR   JNEST
585          JMP   JMP50
586 JMP40    FEVAL FTMP       ;not any token we can use -
587          JSR   CHKCOM     ;try to get a set of x,y coords.
588          FEVAL FTMP2
589 JMP50    LDA   SHOFLG
590          PHA
591          BIT   DIST       ;is this the 1st coord set, and
592          BVC   JMP60      ;specified without a TO clause ?
593          LDA   #0         ;yes-reset flag-this seq. must
594          STA   DIST       ;be executed *once*, only.
595          JSR   HIDE       ;jump to specified coords, but
596 ;                         ;do *not* draw a line.
597 JMP60    FSUB  FTMP,XC,DX  ;Get delta_X & delta_Y
598          FSUB  FTMP2,YC,DY
599          JSR   DRAWLIN    ;     * plot a line *
600          PLA
601          STA   SHOFLG
602          JSR   GETC3
603          CMP   #TOTOKE    ;is there another TO clause ?
604          BEQ   JMP10      ;if so, repeat.
605          RTS
606 JCENTR   LDX   #9          ;copy CENTER coords to XY save
607 JCE10    LDA   CCORDS,X
608          STA   FTMP,X
609          DEX
610          BPL   JCE10
611          LDX   #3
612 JCE20    LDA   ICORDS,X
613          STA   IXC,X
614          DEX
615          BPL   JCE20
616          RTS
617 JNEST    LDX   #9          ;copy NEST coords to XY save
618          LDA   #0
619 JNE10    STA   FTMP,X
620          DEX
621          BPL   JNE10
622          LDX   #3
623 JNE20    STA   IXC,X
624          DEX
625          BPL   JNE20
626          RTS
627 ;
628          COP TURTLE2,D2
```

END OF LISTING 1

continued on page 26

SOME HISTORIC BREAKTHROUGHS DON'T TAKE AS MUCH EXPLAINING AS COMPUSERVE.

But then, some historic breakthroughs could only take you from the cave to the tar pits and back again.

CompuServe, on the other hand, makes a considerably more civilized contribution to your life.

It turns that marvel of the 20th century, the personal computer, into something useful.

Unlike most personal computer products you read about, CompuServe is an information service. It isn't software. It isn't hardware. And you don't even have to know a thing about programming to use it. You subscribe to CompuServe —and 24 hours a day, 7 days a week, it puts a universe of information, entertainment and communications right at your fingertips.

A few of the hundreds of things you can do with CompuServe.

COMMUNICATE

EasyPlex™ Electronic Mail lets even beginners compose, edit, send and file messages the first time they get online. It puts friends, relatives and business associates—anywhere in the country —in constant, convenient touch.

CB Simulator features 72 channels for "talking" with thousands of other enthusiastic subscribers throughout the country and Canada. The chatter is frequently hilarious, the "handles" unforgettable, and the friendships hard and fast.

More than 100 Forums welcome your participation in "discussions" on all sorts of topics. There are Forums for computer owners, gourmet cooks, investors, pilots, golfers, musicians, you name it! Also, Electronic Conferencing lets businesses put heads together without anyone having to leave the shop.

Bulletin Boards let you "post" messages where thousands will see them. You can use our National Bulletin Board or the specialized Bulletin Boards found in just about every Forum.

HAVE FUN

Our full range of games includes "You Guessed It!", the first online TV-style game show you play for real prizes; and MegaWars III, offering the ultimate in interactive excitement. And there are board, parlor, sports and educational games to play alone or against other subscribers throughout the country.

Movie Reviews keep that big night at the movies from being a five star mistake.

SHOP

THE ELECTRONIC MALL℠ gives you convenient, 24-hour-a-day, 7-day-a-week shopping for name brand goods and services at discount prices from nationally known stores and businesses.

SAVE ON TRIPS

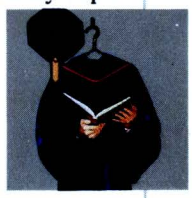
Travelshopper℠ lets you scan flight availabilities (on virtually any airline — worldwide), find airfare bargains and order tickets right on your computer.

Worldwide Exchange sets you up with the perfect yacht, condo, villa, or whatever it takes to make your next vacation *a vacation*.

A to Z Travel/News Service provides the latest travel news plus complete information on over 20,000 hotels worldwide.

MAKE PHI BETA KAPPA

Grolier's Academic American Encyclopedia's Electronic Edition delivers a complete set of encyclopedias right to your living room just in time for today's homework. It's continuously updated ... and doesn't take an inch of extra shelf space.



The College Board, operated by the College Entrance Examination Board, gives tips on preparing for the SAT, choosing a college and getting financial aid.

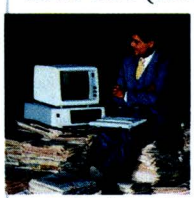
KEEP HEALTHY

Healthnet will never replace a real, live doctor—but it is an excellent and readily available source of health and medical information for the public.

Human Sexuality gives the civilization that put a man on the moon an intelligent alternative to the daily "Advice to the Lovelorn" columns. Hundreds turn to it for real answers.

BE INFORMED

All the latest news is at your fingertips. Sources include the AP news wire (covering all 50 states plus national news), the Washington Post, USA TODAY Update, specialized business and trade publications and more. You can find out instantly what Congress did yesterday; who finally won the game; and what's happening back in Oskaloosa with the touch of a button. And our electronic clipping service lets you tell us what to watch for. We'll electronically find, clip and file news for you...to read whenever you'd like.



INVEST WISELY

Comprehensive investment help just might tell you more about the stock you're looking at than the company's Chairman of the Board knows. (Don't know who he is? Chances are, we can fill you in on that, too.) CompuServe gives you complete statistics on over 10,000 NYSE, AMEX and OTC securities. Historic trading statistics on over 50,000 stocks, bonds, funds, issues and options. Five years of daily commodity quotes. Standard & Poor's. Value Line. And more than a dozen other investment tools.

Site II facilitates business decisions by providing you with demographic and sales potential information by state, county and zip code for the entire country.

National and Canadian business wires provide continuously updated news and press releases on hundreds of companies worldwide.

GET SPECIALIZED INFORMATION

Pilots get personalized flight plans, weather briefings, weather and radar maps, newsletters, etc.

Entrepreneurs use CompuServe too for complete step-by-step guidelines on how to incorporate the IBMs of tomorrow.

Lawyers, doctors, engineers, military veterans and businessmen of all types use similar specialized CompuServe resources pertinent to their unique needs.

And now for the pleasant surprise.

Although CompuServe makes the most of any computer, it's a remarkable value. With CompuServe, you get low start-up costs, low usage charges and local phone-call access in most major metropolitan areas.

Here's exactly how to use CompuServe.

First, relax.

There are no advanced computer skills required.

In fact, if you know how to buy breakfast, you already have the know-how you'll need to access any subject in our system. That's because it's "menu-driven," so beginners can simply read the menus (lists of options) that appear on their screens and then type in their selections.

Experts can skip the menus and just type in "GO" followed by the abbreviation for whatever topic they're after.

In case you ever get lost or confused, just type in "H" for help, and we'll immediately cut in with instructions that should save the day.

Besides, you can either ask questions online through our Feedback service or phone our Customer Service Department.

How to subscribe.

To access CompuServe, you'll need a CompuServe Subscription Kit, a computer, a modem to connect your computer to your phone, and in some cases, easy-to-use communications software. (Check the information that comes with your modem.)

With your Subscription Kit, you'll receive:

- a $25 usage credit.
- a complete hardcover Users Guide.
- your own exclusive user ID number and preliminary password.
- a subscription to CompuServe's monthly magazine, *Online Today.*

Call **800-848-8199** (in Ohio, 614-457-0802) to order your Subscription Kit or to receive more information. Or mail this coupon.

Kits are also available in computer stores, electronic equipment outlets and household catalogs. You can also subscribe with materials you'll find packed right in with many computers and modems sold today.

☐ Please send me additional information.
☐ Please send me a CompuServe Subscription Kit.
☐ I am enclosing my check for $39.95, plus $2.50 handling. *(Add sales tax if delivered in Ohio.)*

Please make check payable to CompuServe Information Services, Inc.

☐ Charge this to my VISA/MasterCard

#_____

Expiration Date _____

Signature _____

Name _____

Address _____

City _____

State _____ Zip _____

MAIL TO:

CompuServe®

Customer Service Ordering Dept.
P.O. Box L-477
Columbus, Ohio 43260

PRI-3701

An H&R Block Company

LISTING 2: TURTLE2 — Source File for MicroSPARC Assembler

```
 0 ;
 1 ;**********************
 2 ;*      TURTLE2       *
 3 ;*    TURTLE BASIC    *
 4 ;*    BY J.B. WARD    *
 5 ;* COPYRIGHT (C) 1986 *
 6 ;* BY MICROSPARC, INC *
 7 ;* CONCORD, MA  01742 *
 8 ;**********************
 9 ;
10 ;  Print a line to hires screen.
11 ;  Syntax: [ HPRINT(x,y) ...line... ]
12 ;          [ HPRINT ...line... ]
13 ;  Where: "...line..." is any line
14 ;  description valid for the "PRINT"
15 ;  statement,except for use of "TAB(n)".
16 ;  x & y are the hires XY coords of the
17 ;  upper left hand corner of the 1st
18 ;  character printed.
19 HPRINT   JSR   GETC1      ;Get next char...
20          CMP   #$28       ;If it's not "(",assume no
21          BNE   HPR030     ;xy coords are  specified.
22          JSR   GETC2      ;(throw left paren away)
23          FEVAL FTMP       ; get x-coord
24          JSR   CHKCOM     ; comma?
25          FEVAL FTMP2      ; get y-coord
26          JSR   CHKRPA     ; right paren?
27          JSR   CHKXY
28          LDA   Y2TMP
29          STA   HPRY
30          LDX   #$FF       ;"divide" X-coord by 7
31          LDA   X2TMP
32 HPR010   INX
33          CMP   #7         ;HPRX := int(x/7)
34          BCC   HPR020     ;HPRX+1 := x mod 7
35          SBC   #7
36          JMP   HPR010
37 HPR020   STX   HPRX
38          STA   HPRX+1
39 HPR030   LDA   CSWL       ;Save character-output vector;
40          STA   CSWSAV     ;Replace it with our addr.
41          LDA   CSWH
42          STA   CSWSAV+1
43          LDA   #HPRDRV
44          STA   CSWL
45          LDA   #HPRDRV/
46          STA   CSWH
47          JSR   GETC3
48          JSR   PRINT2     ;***Call PRINT routine***
49          LDA   CSWSAV
50          STA   CSWL
51          LDA   CSWSAV+1
52          STA   CSWH
53          RTS
54 ;      Fix & check x-y coords
55 CHKXY    FIX   FTMP,X2TMP
56          FIX   FTMP2,Y2TMP
57          LDA   X2TMP+1
58          BEQ   CXY10      ;   -1<x<280 ?
59          CMP   #2
60          BCS   CXY999
61          LDA   X2TMP
62          CMP   #280
63          BCS   CXY999
64 CXY10    LDA   Y2TMP+1    ;  -1<y<192 ?
65          BNE   CXY999
66          LDA   Y2TMP
67          CMP   #192
68          BCS   CXY999
69          RTS
70 CXY999   JMP   ILLQTY     ;("Illegal Quantity")
71 ;  Hires-print character output driver
72 HPRDRV   STA   $D0
73          PHA
74          TXA
75          PHA
76          TYA
77          PHA
78          LDA   $D0
79          AND   #$7F
80          CMP   #$20
81          BCS   HDR015
82          CMP   #$0D       ;Carriage-return?
83          BNE   HDR010
84          JSR   HICR
85          JMP   HDR030
86 HDR010   CMP   #7         ;Bell character?
87          BNE   HDR015
88          JSR   BELL
89          JMP   HDR030
90 HDR015   SBC   #$20
91          STA   $D0
92          LDA   #0
93          ASL   $D0
94          ROL
95          ASL   $D0
96          ROL
```

continued on page 29

JE520 Voice Synthesizer
for Apple II, II+ and *IIe*

JE520AP. . . . $119.95
JE523TS (Software) $ 9.95

JE614 Numeric Auxiliary Keypad
JE614. . . . . $49.95

12" Monochrome Monitor
for Apple II, II+ *IIe* and *IIc*

AMON. $99.95

Joystick
for Apple *IIe*

AJS-1. . . . . $19.95

Joystick
for Apple II, *IIe* and IBM

UJS-1. . . . . $29.95

Add'l. Apple Compatible Products

Cooling Fan
for Apple II, II+ and *IIe*

APF-1. . . . . $39.95

Switching Power Supply
for Apple II, II+ and *IIe*

KHP4007. . . . $39.95

Parallel Printer Interface
for Apple *IIc*

MW-100. . . . . $69.95
DC9200A 9VDC Xformer $3.25

VERSACARD
for Apple II, II+ and *IIe*

VERSACARD. . . . $169.95
PM-SC (Serial Cable) . . . $24.95
PM-PC (Parallel Cable) . . . $24.95

Mail Order Electronics - Worldwide
Jameco ELECTRONICS

Serving the Computer Enthusiast Since 1974

QUALITY COMPUTER PRODUCTS FOR APPLE II, II+, *IIe*, *IIc* and Macintosh*

APPLE* Compatible Cards
FOR APPLE II, II+ AND *IIe**

Jameco ELECTRONICS

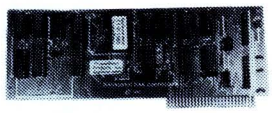
Five Key Software Programs!

128K RAM Card** *RAM Disk!*

The **JE868** is functionally compatible with the Apple II language card and can be utilized with all software that can be used with a standard 16K card. The JE868 requires no modifications to your Apple computer. Five key software programs are included: Memory Management System, utilities, diagnostics, demos, and RAM disk emulators for DOS 3.3, CP/M and Apple Pascal. Also features DOS relocator. Complete with instructions.

JE868 (Expand-A-RAM) . . . $119.95

PROMETHEUS PRODUCTS • INC

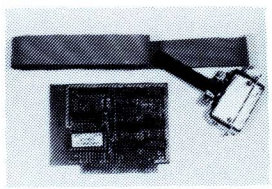


Applesurance Diagnostic Disk Controller Card

The **JE877** serves as a diagnostic tool, an assurance/maintenance tool and a dual disk drive controller. The JE877 will verify and check the operating hardware of your system each time you turn on your Apple II, II+ or *IIe*.* Test your RAM, ROM, CPU, and disk drives. Diagnostic routines may be cancelled at the touch of a key. Complete with instructions.

JE877 (DRV-1/Applesurance II) . . . $69.95

PROMETHEUS PRODUCTS • INC

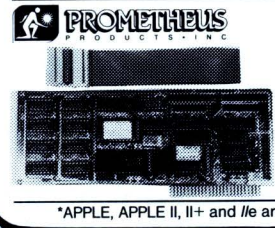


Parallel Printer Card

The **JE880** Printer Interface board is an intelligent interface to most of today's popular dot-matrix graphics printers. The JE880 is fully compatible with Apple CP/M, Apple Pascal (or FORTRAN), and most other operating systems and software packages available for Apple II, II+ and *IIe*.* The JE880 is shipped configured for the Centronics standard and can be re-configured for other standards if necessary. Advanced text printing features include: video screen echo ON or OFF, auto/disable linefeed after carriage return, set/clear the 8th bit of the output data, set left margin and more! Complete with instructions.

JE880 (PRT-1). $59.95

PROMETHEUS PRODUCTS • INC



Parallel/Serial Buffer Card

The **JE883** provides the user with up to 64K of additional or buffered memory (18 pgs. of information). Using the parallel jumper cable supplied, the JE883 will attach to the JE880 (above). Parallel Card needed for operation. The JE883 includes a standard parallel input with both parallel and serial (RS232) buffered outputs. With these features you may access and buffer information to two types of printers (one serial, one parallel) simultaneously. Complete with instructions.

JE883 (P/S Buffer). $79.95

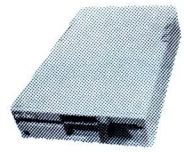
*APPLE, APPLE II, II+ and *IIe* are registered trademarks of Apple Computers.

APPLE™ Compatible 5¼" Disk Drive and Controller Card

• Belt-driven • 143K formatted storage • Color matches Apple Computer • Works with Apple Controller or other Apple-compatible controllers (JE875) • Complete with connector — just plug into your disk controller card • 35 tracks • Size: 6"W x 3½"H x 8-9/16"D • Weight: 4¼ lbs.

ADD-514 (Disk Drive) $139.95
JE875 (Controller Card) $49.95

APPLE™ Compatible 5¼" Half-Ht. Disk Drive



• Direct drive • 143K formatted • 35 tracks • Super quiet • Works with Apple Controllers or other compatibles (JE875) above • Complete with connector-just plug into your controller • Size: 5¾"W x 1⅝"H x 8"D • Weight: 4 lbs.

ADD-12. . . . $129.95

APPLE™ *IIc* Compatible 5¼" Half-Ht. Disk Drive

• Same specs as ADD-12 (above) except no controller necessary.

ADD-IIc. . . . $129.95

80-Col. plus 64K RAM for your APPLE *IIe**

Now you can double the memory capacity and get an 80-column display format for your APPLE *IIe** computer at an affordable price. Just plug the JE864 card into your APPLE* and expand your display to 80 characters per line. Perfect for word processing. The JE864 also features 64K bytes of additional memory to allow programming not possible with standard APPLE *IIe** computers. Complete documentation included.

JE864. . . . . $69.95

16K RAM Card** (Language Card)

The JE860 RAM Card allows the Apple* II and II+ computers to expand from 48K to 64K. Complete with instructions.

JE860. . . . . $39.95

APPLE Keyboard and Case for Apple II and II+

KEYBOARD (KB-A68):
• 68 Keys • 15-Key Keypad • Direct connection with 16-pin ribbon connector • Special features include automatic-key repeat, cursor control/numeric keypad, 46 predefined function keys (Applesoft™, DOS commands and CP/M™), 10 user-definable keys *and more!* • Color (keys): white/grey • Size: 14½"L x 5½"W x 1½"H

CASE (EAEC-1):
• Accommodates KB-A68 • Pop-up lid for easy access • Fits power supply *and* motherboard • Size: 15½"L x 18"D x 4¼"H

Part No.	Description	Price
KB-EA1	Apple Keyboard and Case.	$99.95
KB-A68	68-Key Apple Keyboard only.	$69.95
EAEC-1	Expanded Apple Enclosure Case only.	$34.95

APPLE* Modems and Options

ProModem 1200
for any RS232 Computer
PM1200 (Stand-Alone) $299.95

MAC PAC *for your Macintosh*
Includes: ProModem 1200 external modem, cable and ProCom-M software.

MAC PAC . . . $349.95

OPTIONS for PM1200/MAC PAC

PM-Special #2 Includes Comm. Buffer, 512K Memory and Display. **$249.95**

1200/300 Baud 212A
Telephone Modem for your Apple II, II+ & *IIe*. Just plug in — Hayes compatible!

PM1200A . . . $299.95

300 Baud for *IIc*
Attaches to the rear of Apple *IIc*. Built-in speaker.

PM300c. . . $149.95

$20.00 Min. Order - U.S. Funds Only
Calif. Residents Add 6% or 6½% Sales Tax
Shipping - Add 5% plus $1.50 Insurance
Send S.A.S.E. for Quarterly Sales Flyer!

Send $1.00 Postage for your *FREE 1986 Jameco Catalog*
Prices Subject to Change

MasterCard

Mail Order Electronics - Worldwide
Jameco ELECTRONICS®

VISA®

1355 SHOREWAY ROAD
BELMONT, CALIFORNIA 94002
Phone Orders Welcome (415) 592-8097
Telex No. 176043

1/86

*APPLE, APPLE II, II+, *IIe*, *IIc* and Macintosh are registered trademarks of APPLE Computers

**When using CP/M, the JE860 and JE868 will only function with Versions 2.20 or earlier.

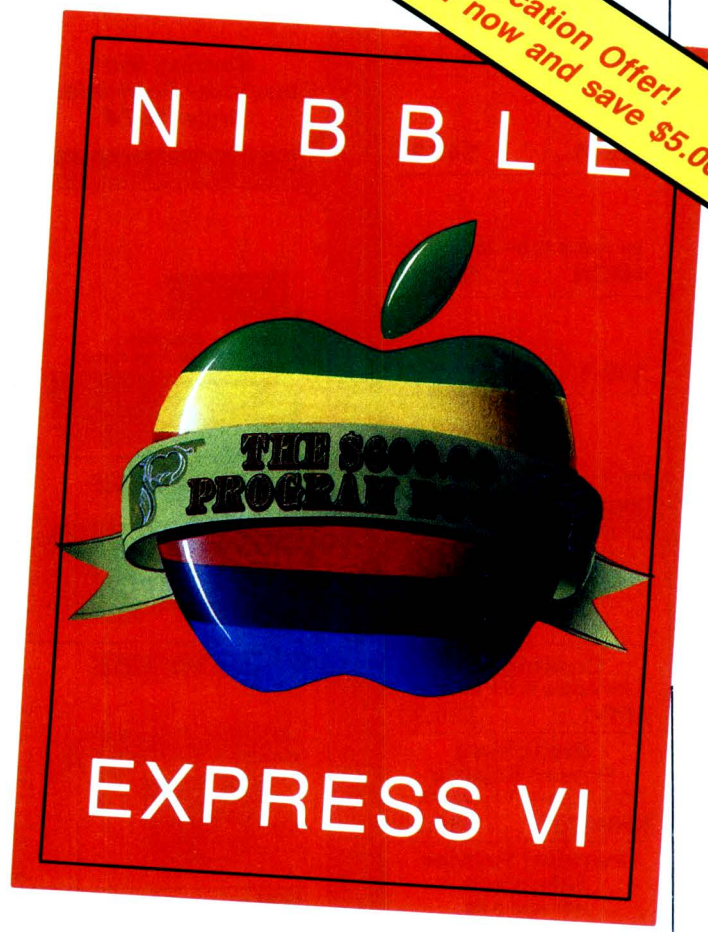
Prepublication Offer!
Order now and save $5.00!

The *Biggest* $600 Apple Program Book

Nibble Express Volume VI is the biggest collection of Apple programs we've ever published! Express VI is 288 pages packed with over $600* of ready-to-type Apple programs from Nibble Magazine 1985.

Even if you've collected all of the 1985 Nibble issues, you'll want Nibble Express VI with its updated listings in one convenient package. And if you place your order before the April 15 publication date, you'll save $5.00 off the $19.95 cover price!

Here's a small sample of what you'll get:



Home/Personal:

NIBBLE ARCHITECT is a powerful program for planning room designs and laying out everything from your garden to your office! Scaled, Hi-Res graphics objects can be easily moved, changed, copied, and printed.

NIBBLE MEDIC helps you keep track of your family's medical expenses and prescription history with a variety of editing and report formats.

APPLE HIGHWAYS will make vacation planning a snap by finding the best route between 170 major U.S. cities. It prints both the route and the mileage between junctions.

* Commercial value of programs if purchased on disk

Programming Utilities:

NAVIGATOR makes program documentation a breeze! It reads your Applesoft programs and prints flowcharts complete with graphic symbols.

APPLESOFT SUPERCHARGER powers your Apple with a set of machine language utilities to format output, control user input, chain programs with common variables, and much more.

SPOOLER keeps your printer busy while you keep computing. It uses the 16K RAM card area to store printer output as you work under DOS 3.3.

Education:

NIBBLE PLANETARIUM is a Hi-Res star charting system with the features of a real planetarium. Display and print the sky from any location in the Northern Hemisphere and at any time.

NIBBLE MAESTRO transforms the Apple keyboard into a four-octave organ. Graphic notes appear on a staff as you play, and you can save tunes to disk and edit or play them later.

Games:

THE NIBBLE 500 challenges you to race for glory through a curving track with random obstacles and a pit stop.

PENGUIN is an arcade-style strategy game with increasing skill levels. Help Sammy the Penguin fend off the invaders to his little island!

☐ ***Yes! I want to save $5.00!***

Send me Nibble Express VI at the prepublication price of $14.95. (Price after April 15, 1986: $19.95)

Also send me:

☐ Nibble Express I — $14.95
☐ Nibble Express II — $14.95
☐ Nibble Express III — $17.95
☐ Nibble Express IV — $17.95
☐ Nibble Express V — $19.95

Please add $1.75 shipping/handling per book. For shipment outside the U.S., add $2.75 for surface mail or $6.50 for air mail per copy. Purchases payable in U.S. funds only. Massachusetts residents add 5% sales tax.

Name _____ Address _____

City _____ State _____ Zip Code _____

☐ I've enclosed a check or money order
☐ Charge my: ☐ Visa ☐ MasterCard

Credit Card Number _____ Expiration Date _____

Signature _____ Telephone Number _____

Send to: Nibble Publications, 45 Winthrop Street, Concord, MA 01742. Or call (617) 371-1660 for ordering with Visa or MasterCard.

LISTING 2: TURTLE2 *(continued)*

```
 97           ASL   $D0
 98           ROL
 99           STA   $D1
100           LDA   $D0
101           ADC   CTABAD
102           STA   $D0
103           LDA   $D1
104           ADC   CTABAD+1
105           STA   $D1
106           LDA   HPRX
107           STA   X1TMP
108           LDY   #7          ;Loop 8 times to put all 8
109           STY   Y2TMP       ;rows of character to screen.
110 HDR020    JSR   HPRADR      ;(get screen address)
111           JSR   HPRPUT      ;(put char.r row to scrn)
112           DEC   Y2TMP
113           BPL   HDR020
114           INC   HPRX        ;Increment char.position
115           LDA   HPRX
116           CMP   #40         ;If at end-of-screen,
117           BCC   HDR030      ;do a <cr> action.
118           JSR   HICR
119 HDR030    PLA
120           TAY
121           PLA
122           TAX
123           PLA
124           RTS
125 ;  Do a hires carriage-return action
126 HICR      LDA   #0
127           STA   HPRX
128           CLC
129           LDA   HPRY
130           ADC   #8
131           CMP   #185
132           BCC   HIC010
133           LDA   #0
134 HIC010    STA   HPRY
135           RTS
136 ;  Get byte address in hires screen
137 ;  for one hires character.
138 HPRADR    CLC
139           LDA   HPRY
140           ADC   Y2TMP
141 HPRYAD    LDX   #0          ;Entry for lineplot y-adr
142           STX   GBASH
143           TAX
144           AND   #$38
145           ASL
146           ASL
147           ASL
148           ROL   GBASH
149           ASL
150           ROL   GBASH
151           STA   GBASL
152           TXA
153           AND   #$C0
154           LSR
155           STA   X1TMP+1
156           LSR
157           LSR
158           ORA   X1TMP+1
159           ORA   GBASL
160           ADC   X1TMP
161           STA   GBASL
162           TXA
163           AND   #$07
164           ASL
165           ASL
166           ORA   GBASH
167           ADC   HRPAGE
168           STA   GBASH
169           RTS
170 ;  Put one line of a character to screen.
171 ;  characters are permitted to cross the
172 ;  40 "character column" boundaries.
173 HPRPUT     LDY   Y2TMP
174           LDA   ($D0),Y     ;get line of char.
175           LDY   #0
176           LDX   HPRX+1
177           BEQ   HPU020      ;-if char.doesnt cross bound.
178           STY   $9D         ;char. does cross...
179 HPU010    ASL               ;align to span 2 addrs.
180           ROL   $9D
181           DEX
182           BNE   HPU010
183           ASL               ;adjust for colour bit (bit7)
184           ROL   $9D
185           LSR
186           JSR   PUTCHR      ;put to screen
187           LDX   HPRX
188           CPX   #39         ;test for RHS of screen.
189           BCS   HPU030
190           INY
191           LDA   $9D
192 HPU020    JSR   PUTCHR      ;put to screen
193 HPU030    RTS
194 ; ****************************************
195 ; *  P L O T T I N G     R O U T I N E S  *
196 ; ****************************************
197 DRAWLIN   LDA   #XC         ;Fix turtle x,y ;If new x,y
198           LDY   #XC/        ;differs from last-plotted x,y
199           JSR   FLOAD       ;by less than 2,then use last
200           JSR   FADHAF      ;plotted x,y to start.
201           JSR   FIX         ;Else use new.
202           SEC
203           LDA   $A1
204           SBC   IXC
205           BPL   DRL10       ;(take abs value)
206           EOR   #$FF
207           CLC
208           ADC   #1
209 DRL10     CMP   #2
210           BCC   DRL20       ;if abs(newx-lastx) >= 2,then:
211           LDA   $A0         ;IXC := fix(XC)
212           STA   IXC+1
213           LDA   $A1
214           STA   IXC
215 DRL20     LDA   #YC
216           LDY   #YC/
217           JSR   FLOAD
218           JSR   FADHAF
219           JSR   FIX
220           SEC
221           LDA   $A1
222           SBC   IYC
223           BPL   DRL30
224           EOR   #$FF
225           CLC
226           ADC   #1
227 DRL30     CMP   #2
228           BCC   DRL40
229           LDA   $A0         ;IYC := fix(YC)
230           STA   IYC+1
231           LDA   $A1
232           STA   IYC
233 DRL40     FIX   FTMP,X1TMP
234           FIX   FTMP2,Y1TMP
235           LDX   #4
236 DRL50     LDA   FTMP,X      ;Update turtle x,y
237           STA   XC,X
238           LDA   FTMP2,X
239           STA   YC,X
240           DEX
241           BPL   DRL50
242           JSR   LPLOT
243           LDX   #3
244 DRL60     LDA   TMPIXC,X
245           STA   IXC,X
246           DEX
247           BPL   DRL60
248           RTS
249 ;       Plot a line on hires screen
250 LPLOT     LDA   #0
251           STA   X1TMP
252           LDA   DX+1
253           STA   XDIR        ;Save sign of dx,dy
254           LDA   DY+1
255           STA   YDIR
256           ABS   DX          ;See if magnitude of dx
257           ABS   DY          ;is greater or less than mag.dy
258           LDX   #0
259 PLT10     LDA   DX,X
260           CMP   DY,X
261           BNE   PLT20
262           INX
263           CPX   #5
264           BCC   PLT10       ;If (mag.dx) >= (mag.dy), plot
265 PLT20     BCS   PLT30       ;y as a function of x; Else plot
266           JSR   FOFY        ;x as a function of y.
267           RTS
268 PLT30     JSR   FOFX
269           RTS
270 ;
271 ;  Find "Y-address"
272 YADR      LDA   TMPIYC+1
273           BPL   YAD10
274           LDA   #0
275           JMP   HPRYAD
276 YAD10     BEQ   YAD20
277           LDA   #191
278           JMP   HPRYAD
279 YAD20     LDA   TMPIYC
280           CMP   #192
281           BCC   YAD30
282           LDA   #191
283 YAD30     JMP   HPRYAD
284 ;       Find "x-address"  ; offset into line & bitmask.
285 XADR      LDA   TMPIXC
286           STA   XSTRT
287           LDA   TMPIXC+1
288           STA   XSTRT+1
289           BPL   XAD10
290           LDA   #0          ;If x<0 , then x := 0
```

continued on next page

LISTING 2: TURTLE2 *(continued)*

```
291           STA   XSTRT
292           STA   XSTRT+1
293           JMP   XAD20
294 XAD10     SEC
295           LDA   XSTRT       ;If x>279 , then x := 279
296           SBC   #279
297           LDA   XSTRT+1
298           SBC   #279/
299           BCC   XAD20
300           LDA   #279
301           STA   XSTRT
302           LDA   #279/
303           STA   XSTRT+1
304 XAD20     ASL   XSTRT       ;Divide x by 7;Quotient is
305           ROL   XSTRT+1     ;byte offset into line ;
306           LDY   #7          ;Remainder is index into
307           LDA   #0          ;bitmask table.
308           STA   XOFSET
309 XAD30     DEY
310           BMI   XAD40
311           ROL   XOFSET
312           ASL   XSTRT
313           ROL   XSTRT+1
314           SEC
315           LDA   XSTRT+1
316           SBC   #7
317           BCC   XAD30
318           STA   XSTRT+1     ;  * XSTRT = MSKIX *
319           BCS   XAD30
320 XAD40     ROL   XOFSET
321           RTS
322 ;       Plot Y as a function of X.
323 FOFX      LDA   DX
324           BNE   FOX20       ;Avoid possible div-by-zero.
325           LDX   #4
326 FOX10     STA   PINC,X
327           DEX
328           BPL   FOX10
329           JMP   FOX30
330 FOX20     FDIV  DY,DX,PINC  ;(dy/dx) used to determine
331 FOX30     LDA   #DX         ;when to inc.or dec.y-coord.
332           LDY   #DX/
333           JSR   PPREP
334 FOX40     JSR   PUTBIT      ; * plot a single point. *
335           JSR   TSTCTR      ; done ?
336           BNE   FOX50       ; ( skip if more to do.)
337           RTS
338 FOX50     JSR   TSTPINC     ;Add test value to test sum;
339           BCC   FOX60       ;Next point up or down 1 bit ?
340           JSR   GNY         ;yes; get-next-Y-address.
341 FOX60     JSR   GNX
342           JMP   FOX40
343 ;       Plot X as a function of Y.
344 FOFY      LDA   DY
345           BNE   FOY20
346           LDX   #4
347 FOY10     STA   PINC,X
348           DEX
349           BPL   FOY10
350           JMP   FOY30
351 FOY20     FDIV  DX,DY,PINC ;(functions same as "FOFX")
352 FOY30     LDA   #DY
353           LDY   #DY/
354           JSR   PPREP
355 FOY40     JSR   PUTBIT      ; * plot a single point *
356           JSR   TSTCTR      ; done ?
357           BNE   FOY50
358           RTS
359 FOY50     JSR   TSTPINC     ;move x up or down by one ?
360           BCC   FOY60       ;(skip if not)
361           JSR   GNX
362 FOY60     JSR   GNY
363           JMP   FOY40
364 ;       Put (plot) a bit on screen.
365 ;    *** Entry for plotting ***
366 PUTBIT    LDA   SHOFLG      ;don't plot if "hiding".
367           BNE   PTB20
368           LDA   TMPIYC+1    ; don't plot if off-screen.
369           BNE   PTB20
370           LDA   TMPIYC
371           CMP   #192
372           BCS   PTB20
373           LDA   TMPIXC
374           SBC   #279
375           LDA   TMPIXC+1
376           BMI   PTB20
377           SBC   #279/
378           BCS   PTB20
379           LDY   XOFSET
380           LDA   PMSK
381 ;     *** Entry for HPRINT ***
382 PUTCHR    BIT   INK
383           BVS   PTB10
384           ORA   (GBASL),Y   ;ink = white
385           STA   (GBASL),Y
386           RTS
387 PTB10     EOR   #$7F        ;ink = black
388           AND   (GBASL),Y
389           STA   (GBASL),Y
390 PTB20     RTS
391 ;       Test for all-points-plotted.
392 TSTCTR    INC   PTCTR+1     ;(reversed due to "fix" call)
393           BNE   TCT10
394           INC   PTCTR
395 TCT10     RTS
396 ;       Get next Y-address
397 GNY       LDA   YDIR
398           BPL   GNY10       ;branch if moving down.
399           SEC
400           LDA   TMPIYC      ;moving up; y := y-1
401           SBC   #1
402           STA   TMPIYC
403           LDA   TMPIYC+1
404           SBC   #0
405           STA   TMPIYC+1
406           JMP   GNY20
407 GNY10     INC   TMPIYC      ;moving down. y := y+1
408           BNE   GNY20
409           INC   TMPIYC+1
410 GNY20     JSR   YADR        ;Get new "y-address"
411           RTS
412 ;       Get next x-address
413 GNX       LDA   XDIR
414           BPL   GNX20
415           SEC
416           LDA   TMPIXC      ;moving left. x := x-1
417           SBC   #1
418           STA   TMPIXC
419           LDA   TMPIXC+1
420           SBC   #0
421           STA   TMPIXC+1
422           SEC
423           LDA   TMPIXC      ;Are we beyond RHS of screen ?
424           SBC   #279
425           LDA   TMPIXC+1
426           SBC   #279/
427           BCS   GNX10
428           LSR   PMSK        ;no...shift bit mask.
429           BCC   GNX10
430           LDA   #$40
431           STA   PMSK
432           DEC   XOFSET
433           BPL   GNX10       ;gone beyond LHS of screen ?
434           LDA   #0          ;yes! pin line at LHS.
435           STA   XOFSET
436           LDA   #1
437           STA   PMSK
438 GNX10     RTS
439 ;                 ....  moving to right.
440 GNX20     INC   TMPIXC
441           BNE   GNX30
442           INC   TMPIXC+1
443 GNX30     LDA   TMPIXC+1
444           BMI   GNX40       ;branch if to left of screen
445           ASL   PMSK
446           BPL   GNX40
447           LDA   #1
448           STA   PMSK
449           INC   XOFSET
450           LDA   XOFSET
451           CMP   #40         ;are we beyond RHS of screen ?
452           BMI   GNX40
453           LDA   #39         ;yes! pin line at RHS
454           STA   XOFSET
455           LDA   #$40
456           STA   PMSK
457 GNX40     RTS
458 PT375     DFC   $80,$40,$00,$00,$00 ; 0.375
459 ;       Prepare temp. areas for plotting a line.
460 PPREP     JSR   FLOAD       ;Get dx or dy;
461           LDA   #PT375
462           LDY   #PT375/
463           JSR   FADD
464           JSR   FIX
465           LDA   $A0
466           EOR   #$FF
467           STA   $A0
468           LDA   $A1
469           EOR   #$FF
470           STA   $A1
471           LDX   #3
472 PPR05     LDA   IXC,X       ;move starting x/y to zero-page
473           STA   TMPIXC,X
474           DEX
475           BPL   PPR05
476 ; Prepare TSTSUM & INCVAL.These values tell us when to
477 ; change the value of the dependent coordinate.
478           LDA   PINC+1      ;In "TSTPINC", INCVAL
479           ORA   #$80        ;is added to a sum, "TSTSUM".
480           STA   INCVAL+1    ;When a carry is generated, we
481           STA   TSTSUM+1    ;must increment the dependent
482           LDA   PINC+2      ;coordinate value.
483           STA   INCVAL
484           STA   TSTSUM
485           LDY   PINC
```

continued on page 32

BackUp Utilities/Boards

E.D.D. III	44.00
CIA Files	47.00
Copy II Plus—Central Point	20.00
Locksmith 5.0—Omega	65.00
Nibbles Away III	59.00
Copy Master II 128K w/Gr. Dump	95.00
Wildcard II—Central Point	75.00

Entertainment

Adventure Construction Set	32.00
7 Cities of Gold or Skyfox	25.00
Murder on the Zinderneuf	14.00
Archon II or One-On-One	25.00
Pinball Construction Set	22.00
Carriers At War or Wilderness	34.00
Reach for the Stars or Bard's Tale	29.00
Europe Ablaze	32.00
Deja Vu	34.00
Racter	28.00
Stephen King's "The Mist"	25.00
James Bond: "A View To Kill"	25.00
Micro League Baseball	24.00
Capt. Goodnight & Islands of Fear	21.00
Gato	23.50
Ghost Busters or Pitfall II	23.50
Hacker or Space Shuttle	25.00
Alaazar or Master of the Lamps	25.00
Wizardry	29.00
Knight Of Diamonds	21.00
Legacy of Llylgamyn	24.00
Return Of Werdna	27.00
Rescue Raiders	22.00
Wisiprint	15.00
Super Zaxxon or Frogger 3 Deep	22.00
The Quest or Ring Quest	20.50
Expedition Amazon, or Xyphus	20.50
Sword of Kadash	20.50
Below The Root	20.00
Swiss Family Robinson	20.00
Millionaire, Tycoon, Baron or Squire	24.00
Solo Flight or Nato Commander	21.00
F-15 Strike Eagle or Acrojet	21.00
Gunship or Silent Service	21.00
Amazon or Dragon World	24.00
Fahrenheit 451 or Shadow Keep	24.00
Flight Simulator II	31.00
Championship Lode Runner	20.50
Choplifter! or Karateka	20.50
Pole Position or Zaxxon / / c	19.00
Bruce Lee, Conan, or Mr. Do	23.50
Murder By The Dozen or Felony	23.00
Kings Quest	30.00
Romancing The Throne	28.00
Beyond Castle Wolfenstein	20.00
Ultima III: Exodus or Ultima IV	34.00
Sargon III	30.00
A Mind Forever Voyaging	27.00
Cutthroats, or Seastalker	23.50
Hitchhiker's Guide to The Galaxy	23.50
Deadline, or Starcross or SpellBreaker	28.50
Witness, Planetfall, or Enchantor	23.50
Infidel or Sorceror	26.50
Zork II, Zork III, or Suspect	26.50
Zork I or Wishbringer	23.50
Operation Market Garden	30.00
Objective Kursk or Phantasie	23.50
War In Russia	47.50
50 Mission Crush or Broadsides	23.50
Kampfgruppe or Reforger 88	35.50
Imperium Galactum	23.50
Germany 1985 or Carrier Force	35.50
Baltic 85 or Gemstone Warrior	20.50
Geopolitique 1990 or Rails West	23.50
North Atlantic '86	35.50
Battle For Normandy	24.00

Utility Software

Apple Mechanic	17.00
Beagle Basic	20.00
Beagle G.P.L.E.	27.00
Beagle Graphics	32.00
D-Code or Extra K	22.00
Disk Quick or I.O. Silver	16.00
Dos Bos or Silicon Salad	13.00
Double Take or Pro-Byter	19.00
Fat Cat	19.00
Flex Text or Frame Up	16.00
Pro-Byter	19.00
Pronto Dos or Utility City	16.00
Triple Dump	21.00
Typefaces	10.00
Bag of Tricks	25.00
Applesoft Compiler (Microsoft)	108.00
Lisa V2.6	53.00
Merlin Pro	57.00
Merlin Assembler	35.00
Merlin Combo Pack	64.00
Apple II Instant Pascal	105.00
Apple Pascal 1.2	100.00
Turbo Pascal 3.0	38.00
Turbo Tool Box	32.00
Visible Computer: 6502	29.00
The Graphics Magician	35.00
New Complete Graphics Sys.	47.00
Fantavision	32.00
Movie Maker	26.00
Dazzle Draw	37.00
Prince (C.Itoh or Epson)	46.00
Pix-It	30.00
Take 1	35.00
Blazing Paddles	30.00
Sideways	35.00
Fontrix 1.5	55.00
Font Paks 1-13 ea.	11.50
DMP Utilities—(Vilberg Bros.)	Call
AppleWorks Modifier 64K	39.00
AppleWorks Modifier 128K	49.00

Blank Media

Elephant SS/DD (10)	11.95
Maxell SS/DD (10)	13.95
Verbatim SS/DD (10)	13.95
BASF SS/DD (10)	10.50
Memorex SS/DD (10)	13.50
3M SS/DD (10)	16.95
Dysan SS/DD (10)	22.95

Business Software

Advanced Logic Systems	
SpellWorks	32.00
Apple Access / / (/ /e, / /c only)	59.00
Apple Works (/ /e, / /c only)	195.00
Apple Writer / / (/ /e, / /c only)	117.00
Quickfile (/ /e only)	75.00
Advanced Logic Systems	
The Handlers (II+, / /e, / /c)	97.00
Applied Software Technology	
Versaform / /e, / /c	45.00
Artsci MagiCalc ProDos	79.00
Magic Office System	125.00
Magic Window II	79.00
BPI Systems	
General Accounting Dos 3.3	199.00
AR, AP, PR or Inv. Dos 3.3 (ea.)	199.00
General Accounting ProDos	219.00
AR/AP/PR or Inv. ProDos (ea.)	219.00
Continental Software F.C.M./FL G/L, A/R, A/P, Payroll ea.	130.00
Home Cataloger	30.00
Dow Jones Software	
Investor's Workshop	89.00
The Market Analyzer Plus	199.00
The Market Microscope	199.00
Dow Jones Membership Kit	19.00
Hayden Software	
Pie Writer V2.2	89.00
Human Edge Software	
Communications Edge	88.00
Management Edge	88.00
Mind Prober	29.00
Mind Over Minors	Call
Sales Edge	88.00
International Solutions	
MouseCalc	89.00
MouseWord	79.00
MouseBudget	41.00
MouseDesk	24.00
Kensington Microware	
Format II Enhanced	88.00
Living Videotext	
Think Tank II+ or / /e	79.00
MECA Managing Your Money	105.00
Megahaus ReportWorks	69.00
MegaWorks	69.00
Microsoft	
Multiplan (Apple Dos Version)	59.00
Microstuff Crosstalk	105.00
Nolo Press Will Writer	29.00
Organizational Software	
Omnis 3	229.00
Paladin Flashcalc	59.00
PBI Software Graphworks 1.2C	59.00
Jeeves	39.00
Peachtree	
Back To Basics: GL/AR/AP	105.00
Pinpoint Publishing Pinpoint	45.00
Practicorp Practicalc II	31.00
Precision Software SuperBase	59.00
Quark Catalyst / /e	95.00
Terminus / /e	55.00
Word Juggler & Lexicheck / /e	65.00
Roger Wagner Publishing	
Mouse Write	75.00
Satellite Software Wordperfect	85.00
Sensible Software	
Bookends—ProDos	79.00
Graphics Department	69.00
The Sensible Speller—Dos or ProDos	69.00
Sierra On-Line	
General Manager II	129.00
Screenwriter Professional	78.00
Software Arts	
The VisiCalc Package	99.00
Software Publishing	
PFS: First Success	189.00
PFS:File, Graph, Report	69.00
PFS:Write or Plan (/ /e, / /c)	69.00
PFS:Access (/ /e, / /c)	42.00
Sorcim/IUS SuperCalc 3a	105.00
State of the Art	
Electric Checkbook	59.00
GL/AR/AP/Inventory/Payroll	289.00
Stoneware	
DB Master Business Team	189.00
DB Master Business Writer	89.00
DB Master Version Four Plus	169.00

Educational Software

Apple	
Apple Logo / / 128K (/ /e, / /c only)	79.00
Baron's Computer SAT	53.00
CBS Software Adventure Master	26.00
America Coast to Coast	24.00
Dinosaur Dig	24.00
Forecast or T-Rex	28.00
Goren's Bridge	49.00
Mastering The SAT	59.00
Micro Speed Reading	59.00
Success w/Algebra (ea.)	21.00
Success w/Math (ea.)	15.00
Davidson & Associates	
Alge-Blaster	29.00
Classmate or Math Blaster	29.00
Speed Reader II	41.00
Speed Reader Data A, B, C, or D	12.00
Spell-It or Word Attack!	29.00
Designware Body Transparent	26.00
European Nations & Locations	26.00
Grammer Examiner	26.00
Remember	47.00
States & Traits	26.00
DLM	
Alligator Mix or Demolition Division	20.25
Freddie's Puzzles or Number Farm	19.00
Spelling Wiz or Word Invasion	20.25
Harcourt Brace Computer SAT	46.00
Learning Company	
Addition Magician	21.00
Bumble Plot or Reader Rabbit	24.00
Gertrude's Puzzles or Secrets	26.00
Juggle's Rainbow	18.00
Magic Spell or Word Spinner	21.00
Robot Odyssey or Rocky's Boots	29.00
MECC All Titles in Stock!	Call
Peachtree/Eduware	
Algebra 1, 2, 3, or 4	22.00
Algebra 5 or 6	27.00
Compu-Read	17.00
Compu-Math: Arithmetic Skills	29.00
Introduction to Counting	24.00
PSAT or SAT Word Attack Skills	17.00
PSAT/SAT Analogy	18.00
Spelling Bee w/Reading Primer	23.00
Writing Skills 1, 2, 3, 4 or 5	24.00
Scarborough Systems	
Mastertype's Writing Wizard	48.00
Build A Book About You	24.00
Original Boston Computer Diet	48.00
Run for the Money or SongWriter	24.00
Scholastic Wizware	
Agent U.S.A. or Turtle Tracks	24.00
Operation: Frog or Logic Builders	24.00
Logo Robot or Kids at Work	15.00
Simon & Schuster	
JK Lasser Money Manager	59.00
Spinnaker	
Adventure Creator or Trains	21.00
Aerobics or Snooper Troops	20.00
Counting Parade or Sum Ducks	21.00
Delta Drawing	28.00
File & Report or Spreadsheet	35.00
Fraction Fever	15.00
Grandma's House	15.00
In Search Of....	20.00
Kidwriteror Alphabet Zoo	18.00
Kindercomp or Kids on Keys	15.00
Treasure Island or Wizard of Oz	19.00
Word Processor	35.00
Springboard	
Early Games or Music Maestro	21.00
Fraction Factory or Make a Match	18.00
Rainbow Painter or Puzzle Master	21.00
Newsroom	29.00
Newsroom Clip Art Collection	17.00
Terrapin Terrapin Logo	57.00
TimeWorks	
Evelyn Wood Dynamic Reader	41.00
Xerox Chivalry	32.00
Exploring Tables & Graphs 1 or 2	23.00
Pic.Builder	24.00
Stickybear Series (ea.)	24.00

Modems

Volksmodem 12	169.00
Anchor Signalman Express	239.00
Apple Personal Modem	289.00
Hayes Micromodem / /e	129.00
Hayes 300 Baud Smartmodem / /c	155.00
Hayes 300 Baud Smartmodem	129.00
Hayes 1200 Baud Smartmodem	365.00
Hayes 2400 Baud Smartmodem	579.00
Novation 2400 Baud Standalone Modem	479.00
Novation Apple Cat II	189.00
Novation 212 Apple Cat II	359.00
Popcom X-100 (1200 Baud)	255.00
Pro Modem 1200 (External)	289.00
Pro Modem 1200A (Internal)	259.00
Pro Modem 300c	129.00
U.S. Robotics Password 1200	205.00
U.S. Robotics Courier 2400	389.00
Zoom/Modem / /e	115.00
Zoom/Modem / /e Plus	125.00

Z80/80 Col./Ramcards

Applied Engineering: RamWorks	Call
Applied Engineering: Z-RAM / /c	Call
Checkmate—Multiview 80/160	219.00
Checkmate—Multiram Cards	Call
Quadram Multicore Card	Call
Quadram E-Ram 80	99.00
Apricorn 16K Ramcard II+	68.00
Saturn 128K Ram	175.00
Neptune 80 Col. 64K Card	185.00
Apricorn 64K Extend-It	89.00
Extend 80 (Caribbean Computer)	79.00
Micro Sci 80/64e	69.00
Videx Videoterm w/SS Inv.	178.00
Videx Ultraterm	189.00
ALS Z-Engine 2.2	125.00
Microsoft Premium Softcard / /e	249.00
Microsoft Softcard II	269.00
MTC Speed Demon	159.00
Accellerator / /e	205.00

Interface Cards

Apple Super Serial Card	109.00
SeriAll Interface Card	105.00
AST Multi I/O	155.00
Serial Grappler Plus	75.00
Grappler+	74.00
Buffered Grappler+	135.00
Grappler / /c	78.00
Nice Print (Spies Labs)	109.00
Finger Print Plus II+, / /e, / /c	89.00
Print-It! (Parallel or Serial)	129.00
Pkaso/U Interface	115.00
Video 7 RGB Interface / /c	79.00
Video 7 64K RGB Interface / /e	109.00
Video 7 Color Enhancer / /e, / /c	Call
Video 7 Screen Enhancer / /e, / /c	Call

Accessories

Kensington System Saver	60.00
Cool+Time	69.00
Curtis Diamond	30.00
Curtis Emerald	36.00
Curtis Ruby	54.00
Curtis Sapphire	47.00
Apple Mouse / /e	115.00
Apple Mouse / /c	79.00
Time Trax / /e or / /c Version	79.00
Time Trax / /c System Clock Only	55.00
Hayes Mach II Joystick	27.00
Hayes Mach III Joystick	33.00
Kraft Quickstick / /c	49.00
Kraft Joystick	31.00
Gibson Light Pen	149.00
Digital Paintbrush System	175.00
Muppet Learning Keys	42.00
Koala Pad Plus	72.00
Key Tronic Keyboard	225.00
Trackhouse Keypad	79.00
Mockingboard A Board	71.00
Mockingboard C Board	121.00
Mockingboard D Board (/ /c)	131.00
Echo II Plus	94.00
The Cricket / /c	115.00
Thunderclock Plus	105.00
Hanzon Universal Data Buffer	289.00

Printers

Apple	
ImageWriter II Color Printer	459.00
ImageWriter (Wide Carriage)	475.00
Scribe Printer	199.00
Brother HR-15XL	339.00
HR-25	459.00
HR-35	679.00
TwinWriter 5	859.00
Fortis/Dynax	
DH45 Dual Head Printer	859.00
Citizen MSP-10 (160 cps)	249.00
MSP-15 (160 cps)	369.00
MSP-20 (200 cps)	399.00
MSP-25 (200 cps)	509.00
C.Itoh ProWriter 8510 SP Plus	329.00
ProWriter 1500 SP Plus	469.00
Epson	
LX-80/90	Call 209.00/259.00
FX-85	For Call
FX-185	Most Call
DX-10/20	Recent Call
JX-80	Low Low! Call
LQ-800-1000	Prices! Call
LQ-1500	Call
SQ-2000	Call
Hewlett-Packard ThinkJet	Call
LazarJet	Call
Juki 6000 Par. or Ser.	179.00
6100 Par.	339.00
6200 Par.	509.00
6300 Par. or Ser.	659.00
5520 Par. Color Printer	419.00
Mannesaman Tally	
MT-85 Apple	345.00
MT-86 Apple	415.00
NEC SpinWriter e.l.f. 360	379.00
SpinWriter 2030P (20 cps)	625.00
SpinWriter 3530P (35 cps)	989.00
PinWriter P2-2P (180 cps)	495.00
PinWriter P3-3P (180 cps)	695.00
Okidata Okimate 20 Color Printer	120.00
Microline 182P	215.00
Microline 183P	379.00
Microline 192P	339.00
Microline 193P	475.00
Microline 192 Apple	369.00
Microline 193 Apple	549.00
Panasonic KXP-1091	229.00
KXP-1092	349.00
KXP-1093	409.00
KXP-3151	379.00
Silver Reed EXP-400P	209.00
EXP-500P	239.00
EXP-550P	369.00
EXP-770P	679.00
Star Micronics SG-10 (120 cps)	219.00
SG-15 (120 cps)	379.00
SD-10 (160 cps)	339.00
SD-15 (160 cps)	449.00
SR-10 (200 cps)	489.00
SR-15 (200 cps)	585.00
Toshiba P-1340 Parallel	439.00
P-341 Parallel	899.00
P-1351 Parallel	989.00
P-351 Parallel	1039.00

Disk Drives

Apple	
ProFile (5 Meg) w/Accessory Kit	779.00
ProFile (10 Meg) w/Access. Kit	1039.00
Apple / /c External Drive	229.00
Apple DuoDisk w/Accessory Kit	449.00
Apple UniDisk w/Controller	269.00
Corvus Systems	
OmniDrive Hard Disks	Call
Gamma FD 100c	169.00
Micro Sci A2 Full Height	159.00
A5 Slimline / /e	159.00
A5 Slimline / /c	165.00
Micro Sci C2 Controller	49.00
Mitac AD-3C Slimline / /e or / /c	129.00
Pico Products	
Apple Half-Height II+,/ /e, / /c	139.00
Quark	
QC-10 Hard Disk / /e, / /c or Mac	1029.00
QC-20 Hard Disk / /e, / /c or Mac	1379.00

Monitors

Amdek	
Video 300G 12" Green Monitor	115.00
Video 300A 12" Amber Monitor	119.00
Color Monitor 300 (Composite)	219.00
Color Monitor 500 (RGB)	255.00
Color Monitor 600 (RGB)	385.00
DVM III RGB Interface	119.00
Apple Color Monitor / /e	289.00
Color Monitor / /c	289.00
Panasonic TR120M1 Hi-Res Green	110.00
TR120MD Hi-Res Amber	110.00
DTS101 Dual Mode Color 10"	165.00
DTH103 10" Super Hi-Res RGB	369.00
DTM140 14" Super Hi-Res RGB	319.00
Princeton Graphics HX-9	409.00
HX-12 RGB	439.00
RGB-80 Interface	145.00
Teknika MJ-22 RGB Color Monitor	289.00
Sakata SG-1000 Green	105.00
SA-1000 Amber	115.00
SC-100 Composite Color	149.00
SC-200 RGB Color	379.00
Taxan 115 12" Hi-Res Green	109.00
116 12" Hi-Res Amber	115.00
220 14" Composite Color	215.00
610 12" Hi-Res RGB Color (510x200)	309.00
620 12" Hi-Res RGB Color (640x200)	359.00
410-80/64 RGB-II Interface	169.00

Home Software

Home Accountant	39.00
Dollars and Sense (II+, / /e)	58.00
Dollars and Sense (/ /c)	68.00
Monogram Forecast (64K or 128K)	38.00
Time Is Money	59.00
Money Street	75.00
Networth	48.00
Financial Cookbook	32.00
Cut and Paste	24.00
The Write Choice	27.00
Bank Street Writer	41.00
Bank Street Speller	41.00
Bank Street Filer	41.00
Homeword	42.00
Homeword Speller	31.00
ASCII: Express Professional	76.00
Softerm II	124.00
Compuserve Starter Kit	15.00
Typing Tutor II	15.00
Typing Tutor III	31.00
Master Type	24.00
Crossword Magic	31.00
Micro Cookbook (II+, / /e)	23.00
Music Construction Set	25.00
Print Shop	28.00
Print Shop Companion	25.00
Print Shop Graphics Library I or II	15.00
Rainbow Paper Pak (300 Sheets)	12.00

Ordering Instructions

VISA AND MASTERCHARGE ACCEPTED
For fast delivery send cashier's check, certified check or money order. Fortune 1000 & Government checks honored immediately—Personal and other company checks allow 3 weeks to clear. Mastercard & Visa include card # & expiration date. Connecticut residents add 7.5% sales tax. We ship same day for most orders.

Shipping Charges: Software—$2.50 Minimum charge U.S. Mainland. With C.O.D. shipments add an additional $1.90 U.S. postal, APO, and FPO add 3% ($5.00 Min.). Alaska or Hawaii add 6% ($10.00 Min.). Canadian orders add 10% ($15.00 Min.). Foreign Orders except Canada—Add 18% ($25.00 Min.).

Hardware—please call for shipping charges.
Note: We do not guarantee machine compatibility. All products are new and include factory warranty, therefore ALL sales are final. Defective software will be replaced by the same item only. Defective hardware will be replaced or repaired at our discretion. Prices & availability subject to change without notice. Products purchased in error, subject to 15% restocking fee. All returns must have an authorization number. Call (203) 378-3662 to obtain one before returning defective products for replacement.

For items not listed just call:

Visit Our New Showroom: Exit 32 off I-95

PROGRAMS PLUS

Visit Our New Showroom: Exit 32 off I-95
Stationhouse Square, 2505 Main Street
Stratford, Connecticut 06497
Mon.-Fri. 9-9; Sat. 10-9
Inquiries & Conn. residents call:
(203) 378-3662 or 378-8293

ORDERS ONLY TOLL FREE **1-800-832-3201**

MACINTOSH COMPUTERS . . . Call
APPLE / /c, / /e COMPUTERS Call

Up to 60% off retail

CIRCLE NUMBER 10

LISTING 2: TURTLE2 *(continued)*

```
            CPY   #$81
            BNE   PPR07
            LDA   #$FF
            STA   INCVAL
            STA   INCVAL+1
            STA   TSTSUM
            STA   TSTSUM+1
            JMP   PPR20
PPR07       CPY   #$71
            BCS   PPR10
            LDA   #0
            STA   INCVAL
            STA   INCVAL+1
            JMP   PPR20
PPR10       CPY   #$80
            BCS   PPR20
            LSR   INCVAL+1
            ROR   INCVAL
            INY
            JMP   PPR10
PPR20       JSR   YADR        ;Get initial XY coordinates.
            JSR   XADR
            LDY   MSKIX
            LDA   BMSKS,Y
            STA   PMSK
            RTS
BMSKS       DFC   $01,$02,$04,$08
            DFC   $10,$20,$40
;       Test to see if x or y coord should be altered.
TSTPINC     CLC
            LDA   TSTSUM
            ADC   INCVAL
            STA   TSTSUM
            LDA   TSTSUM+1
            ADC   INCVAL+1
            STA   TSTSUM+1
            RTS
;
            DFS   .^$FFF8+8-. ; align char. table
            ULT
;
;       Hires character table
CHRTAB      DFC   $00,$00,$00,$00,$00,$00,$00,$00 ; blank 20
            DFC   $08,$08,$08,$08,$00,$00,$08,$00 ; !     21
            DFC   $14,$14,$14,$00,$00,$00,$00,$00 ; "     22
            DFC   $14,$14,$3E,$14,$3E,$14,$14,$00 ; #     23
            DFC   $08,$3C,$0A,$1C,$28,$1E,$08,$00 ; $     24
            DFC   $06,$26,$10,$08,$04,$32,$30,$00 ; %     25
            DFC   $04,$0A,$0A,$04,$2A,$12,$2C,$00 ; &     26
            DFC   $08,$08,$08,$00,$00,$00,$00,$00 ; '     27
            DFC   $08,$04,$02,$02,$02,$04,$08,$00 ; (     28
            DFC   $08,$10,$20,$20,$20,$10,$08,$00 ; )     29
            DFC   $08,$2A,$1C,$08,$1C,$2A,$08,$00 ; *     2A
            DFC   $00,$08,$08,$3E,$08,$08,$00,$00 ; +     2B
            DFC   $00,$00,$00,$00,$08,$08,$04,$00 ; ,     2C
            DFC   $00,$00,$00,$3E,$00,$00,$00,$00 ; -     2D
            DFC   $00,$00,$00,$00,$00,$00,$08,$00 ; .     2E
            DFC   $00,$20,$10,$08,$04,$02,$00,$00 ; /     2F
            DFC   $1C,$22,$32,$2A,$26,$22,$1C,$00 ; 0     30
            DFC   $08,$0C,$08,$08,$08,$08,$1C,$00 ; 1     31
            DFC   $1C,$22,$20,$18,$04,$02,$3E,$00 ; 2     32
            DFC   $3E,$20,$10,$18,$20,$22,$1C,$00 ; 3     33
            DFC   $10,$18,$14,$12,$3E,$10,$10,$00 ; 4     34
            DFC   $3E,$02,$1E,$20,$20,$22,$1C,$00 ; 5     35
            DFC   $38,$04,$02,$1E,$22,$22,$1C,$00 ; 6     36
            DFC   $3E,$20,$10,$08,$04,$04,$04,$00 ; 7     37
            DFC   $1C,$22,$22,$1C,$22,$22,$1C,$00 ; 8     38
            DFC   $1C,$22,$22,$3C,$20,$10,$0E,$00 ; 9     39
            DFC   $00,$00,$08,$00,$08,$00,$00,$00 ; :     3A
            DFC   $00,$00,$08,$00,$08,$08,$04,$00 ; ;     3B
            DFC   $10,$08,$04,$02,$04,$08,$10,$00 ; <     3C
            DFC   $00,$00,$3E,$00,$3E,$00,$00,$00 ; =     3D
            DFC   $04,$08,$10,$20,$10,$08,$04,$00 ; >     3E
            DFC   $1C,$22,$10,$08,$08,$00,$08,$00 ; ?     3F
;
            DFC   $1C,$22,$2A,$3A,$1A,$02,$3C,$00 ; @     40
            DFC   $08,$14,$22,$22,$3E,$22,$22,$00 ; A     41
            DFC   $1E,$22,$22,$1E,$22,$22,$1E,$00 ; B     42
            DFC   $1C,$22,$02,$02,$02,$22,$1C,$00 ; C     43
            DFC   $1E,$22,$22,$22,$22,$22,$1E,$00 ; D     44
            DFC   $3E,$02,$02,$1E,$02,$02,$3E,$00 ; E     45
            DFC   $3E,$02,$02,$1E,$02,$02,$02,$00 ; F     46
            DFC   $3C,$02,$02,$02,$32,$22,$3C,$00 ; G     47
            DFC   $22,$22,$22,$3E,$22,$22,$22,$00 ; H     48
            DFC   $1C,$08,$08,$08,$08,$08,$1C,$00 ; I     49
            DFC   $20,$20,$20,$20,$20,$22,$1C,$00 ; J     4A
            DFC   $22,$12,$0A,$06,$0A,$12,$22,$00 ; K     4B
            DFC   $02,$02,$02,$02,$02,$02,$3E,$00 ; L     4C
            DFC   $22,$36,$2A,$2A,$22,$22,$22,$00 ; M     4D
            DFC   $22,$22,$26,$2A,$32,$22,$22,$00 ; N     4E
            DFC   $1C,$22,$22,$22,$22,$22,$1C,$00 ; O     4F
            DFC   $1E,$22,$22,$1E,$02,$02,$02,$00 ; P     50
            DFC   $1C,$22,$22,$22,$2A,$12,$2C,$00 ; Q     51
            DFC   $1E,$22,$22,$1E,$0A,$12,$22,$00 ; R     52
            DFC   $1C,$22,$02,$1C,$20,$22,$1C,$00 ; S     53
            DFC   $3E,$08,$08,$08,$08,$08,$08,$00 ; T     54
            DFC   $22,$22,$22,$22,$22,$22,$1C,$00 ; U     55
            DFC   $22,$22,$22,$22,$14,$14,$08,$00 ; V     56
            DFC   $22,$22,$22,$2A,$2A,$36,$22,$00 ; W     57
            DFC   $22,$22,$14,$08,$14,$22,$22,$00 ; X     58
            DFC   $22,$22,$14,$08,$08,$08,$08,$00 ; Y     59
            DFC   $3E,$20,$10,$08,$04,$02,$3E,$00 ; Z     5A
            DFC   $3C,$04,$04,$04,$04,$04,$3C,$00 ; [     5B
            DFC   $00,$02,$04,$08,$10,$20,$00,$00 ; \     5C
            DFC   $1E,$10,$10,$10,$10,$10,$1E,$00 ; ]     5D
            DFC   $08,$1C,$2A,$08,$08,$08,$00,$00 ; ^     5E
            DFC   $00,$00,$00,$00,$00,$00,$3E,$00 ; _     5F
;
            DFC   $00,$00,$00,$00,$00,$00,$00,$00 ; blank 60
            DFC   $00,$00,$0E,$10,$1C,$12,$2C,$00 ; a     61
            DFC   $02,$02,$1A,$26,$22,$22,$1E,$00 ; b     62
            DFC   $00,$00,$1C,$22,$02,$02,$3C,$00 ; c     63
            DFC   $20,$20,$2C,$32,$22,$22,$3C,$00 ; d     64
            DFC   $00,$00,$1C,$22,$3E,$02,$3C,$00 ; e     65
            DFC   $18,$24,$04,$0E,$04,$04,$04,$00 ; f     66
            DFC   $00,$00,$3C,$22,$22,$3C,$20,$1E ; g     67
            DFC   $02,$02,$1A,$26,$22,$22,$22,$00 ; h     68
            DFC   $08,$00,$0C,$08,$08,$08,$1C,$00 ; i     69
            DFC   $10,$00,$10,$10,$10,$10,$12,$0C ; j     6A
            DFC   $02,$02,$22,$12,$0A,$16,$22,$00 ; k     6B
            DFC   $0C,$08,$08,$08,$08,$08,$1C,$00 ; l     6C
            DFC   $00,$00,$16,$2A,$2A,$2A,$2A,$00 ; m     6D
            DFC   $00,$00,$1A,$24,$24,$24,$24,$00 ; n     6E
            DFC   $00,$00,$1C,$22,$22,$22,$1C,$00 ; o     6F
            DFC   $00,$00,$1E,$22,$22,$1E,$02,$02 ; p     70
            DFC   $00,$00,$3C,$22,$22,$3C,$20,$20 ; q     71
            DFC   $00,$00,$1A,$26,$02,$02,$02,$00 ; r     72
            DFC   $00,$00,$3C,$02,$1C,$20,$1E,$00 ; s     73
            DFC   $00,$04,$0E,$04,$04,$04,$18,$00 ; t     74
            DFC   $00,$00,$12,$12,$12,$12,$2C,$00 ; u     75
            DFC   $00,$00,$22,$22,$22,$14,$08,$00 ; v     76
            DFC   $00,$00,$22,$22,$2A,$2A,$14,$00 ; w     77
            DFC   $00,$00,$22,$14,$08,$14,$22,$00 ; x     78
            DFC   $00,$00,$22,$22,$22,$3C,$20,$1E ; y     79
            DFC   $00,$00,$3E,$10,$08,$04,$3E,$00 ; z     7A
            DFC   $10,$08,$08,$04,$08,$08,$10,$00 ; curly [ 7B
            DFC   $00,$08,$08,$08,$08,$08,$08,$00 ; vertbar 7C
            DFC   $04,$08,$08,$10,$08,$08,$04,$00 ; curly ] 7D
            DFC   $00,$00,$04,$2A,$10,$00,$00,$00 ; tilde   7E
            DFC   $7F,$7F,$7F,$7F,$7F,$7F,$7F,$7F ; rubout  7F
;
```

END OF LISTING 2

LISTING 3: TURTMAC — Source File for MicroSPARC Assembler

```
;
;**********************
;*       TURTMAC      *
;*    TURTLE BASIC    *
;*    BY J.B. WARD    *
;* COPYRIGHT (C) 1986 *
;* BY MICROSPARC, INC *
;* CONCORD, MA 01742  *
;**********************
;
; Macro definitions--also uses MACLIB
;
; Evaluate a flt.pt. expression
FEVAL       MAC
            JSR $DD67
            LDX #:A
            LDY #:A/
            JSR $EB2B
            EMC
; Take abs. value flt.pt. number
ABS         MAC
            LDA :A+1
            AND #$7F
            STA :A+1
            EMC
; Change sign of f.p. number
CHS         MAC
            LDA :A+1
            EOR #$80
            STA :A+1
            EMC
; Load Flt.Pt.Accum#1 with f.p.number
FLOAD       MAC
            LDA #:A
            LDY #:A/
            JSR $EAF9
            EMC
; Save Flt.Pt.Accum#1 to 5-byte field
FSAVE       MAC
            LDX #:A
            LDY #:A/
            JSR $EB2B
            EMC
; Convert floating point to integer
; FIX fl,il    :  il := fix(fl)
FIX         MAC
            FLOAD :A
            JSR $E10C
```

```
 48           LDA $A1
 49           STA :B
 50           LDA $A0
 51           STA :B+1
 52           EMC
 53 ; Add two f.p. numbers
 54 ; FADD f1       : facc1 := f1+facc1
 55 ; FADD f1,f2    : f1 := f1+f2
 56 ; FADD f1,f2,f3 : f3 := f1+f2
 57 FADD      MAC
 58           AIF :B/
 59           FLOAD :A
 60           LDA #:B
 61           LDY #:B/
 62           JSR $E7BE
 63           AIF :C/
 64           FSAVE :C
 65           ALS
 66           FSAVE :A
 67           AEN
 68           ALS
 69           LDA #:A
 70           LDY #:A/
 71           JSR $E7BE
 72           AEN
 73           EMC
 74 ; Subtract two f.p. numbers
 75 ; FSUB f1       : facc1 := f1-facc1
 76 ; FSUB f1,f2    : f1 := f1-f2
 77 ; FSUB f1,f2,f3 : f3 := f1-f2
 78 FSUB      MAC
 79           AIF :B/
 80           FLOAD :B
 81           LDA #:A
 82           LDY #:A/
 83           JSR $E7A7
 84           AIF :C/
 85           FSAVE :C
 86           ALS
 87           FSAVE :A
 88           AEN
 89           ALS
 90           LDA #:A
 91           LDY #:A/
 92           JSR $E7A7
 93           AEN
 94           EMC
 95 ; Multiply two f.p. numbers
 96 ; FMPY f1       : facc1 := f1*facc1
 97 ; FMPY f1,f2    : f1 := f1*f2
 98 ; FMPY f1,f2,f3 : f3 := f1*f2
 99 FMPY      MAC
100           AIF :B/
101           FLOAD :A
102           LDA #:B
103           LDY #:B/
104           JSR $E97F
105           AIF :C/
106           FSAVE :C
107           ALS
108           FSAVE :A
109           AEN
110           ALS
111           LDA #:A
112           LDY #:A/
113           JSR $E97F
114           AEN
115           EMC
116 ; Divide two f.p. numbers
117 ; FDIV f1       : facc1 := f1/facc1
118 ; FDIV f1,f2    : f1 := f1/f2
119 ; FDIV f1,f2,f3 : f3 := f1/f2
120 FDIV      MAC
121           AIF :B/
122           FLOAD :B
123           LDA #:A
124           LDY #:A/
125           JSR $EA66
126           AIF :C/
127           FSAVE :C
128           ALS
129           FSAVE :A
130           AEN
131           ALS
132           LDA #:A
133           LDY #:A/
134           JSR $EA66
135           AEN
136           EMC
```

END OF LISTING 3

LISTING 4: TURTLE

```
8800- 20 8B 88 20 A2 88 20 4B
8808- D6 20 2C 89 20 39 89 4C
8810- 03 E0 F2 8A DD 8A 60 8B
8818- 7C 8B C2 8B CB 8B 38 8C
8820- 5D 8C 4C 8B 94 8C E8 8B
8828- EF 8B D4 8B F6 8B 70 8C
8830- 73 8C 48 8D D7 8A D1 8A
8838- 46 57 C4 42 41 43 CB 4C
8840- 45 46 D4 52 49 47 48 D4
8848- 53 48 4F D7 48 49 44 C5
8850- 43 45 4E 54 45 D2 4E 45
8858- 53 D4 54 55 52 4E 54 CF
8860- 4A 55 4D D0 53 50 4C 49
8868- D4 46 55 4C CC 48 49 52
8870- 45 D3 56 49 45 57 A3 A1
8878- 44 CF 48 50 52 49 4E D4
8880- 42 4C 41 43 CB 57 48 49
8888- 54 C5 00 A0 00 84 50 A9
8890- D0 85 51 AD 81 C0 B1 50
8898- 91 50 C8 D0 F9 E6 51 D0
88A0- F5 60 A9 4C 8D 0B D6 8D
88A8- 37 D7 8D 46 D8 8D E5 D5
88B0- 8D D5 DE A9 A8 8D 0C D6
88B8- A9 89 8D 0D D6 A9 85 8D
88C0- 38 D7 A9 89 8D 39 D7 A9
88C8- C6 8D 47 D8 A9 89 8D 48
88D0- D8 A9 E4 8D E6 D5 A9 89
88D8- 8D E7 D5 A9 F4 8D D6 DE
88E0- A9 89 8D D7 DE A2 06 BD
88E8- 25 89 9D 09 D1 CA 10 F7
88F0- A9 07 8D 20 D0 A9 8C 8D
88F8- 21 D0 A9 17 8D 22 D0 A9
8900- 8C 8D 23 D0 A9 DE 8D 12
8908- D0 A9 8B 8D 13 D0 A9 FF
8910- 8D 83 FD A0 05 B9 66 89
8918- 99 6C 89 88 10 F7 AD 80
8920- C0 AD 80 C0 60 50 4C 54
8928- A3 43 4C D2 A9 FF 85 73
8930- 85 6F A9 87 85 74 85 70
8938- 60 A9 FF 85 E4 A9 00 8D
8940- 77 8A A9 20 85 E6 20 39
8948- 8C 20 D2 8A A9 C8 8D 8B
8950- 8A A9 91 8D 8C 8A A9 7C
8958- 8D F2 03 A9 89 8D F3 03
8960- 49 A5 8D F4 03 60 7B 89
8968- 7B 89 E1 89 00 00 00 00
8970- 00 00 6C 6C 89 6C 6E 89
8978- 6C 70 89 60 AD 80 C0 AD
8980- 80 C0 4C D0 9D C9 FF D0
8988- 05 20 72 89 90 17 C9 EB
8990- B0 06 E9 7E AA 4C 3A D7
8998- E9 EB AA 84 85 A9 38 85
89A0- 9D A9 87 85 9E 4C 44 D7
89A8- A5 0F C9 E9 90 0B 20 75
89B0- 89 90 0E BD 00 02 4C 0E
89B8- D6 A9 37 85 9D A9 88 85
89C0- 9E A0 00 4C A8 D5 C9 FF
89C8- D0 03 4C 78 89 38 E9 6C
89D0- C9 13 B0 0D 0A A8 B9 13
89D8- 88 48 B9 12 88 48 4C B1
89E0- 00 4C C9 DE F0 07 C9 48
89E8- F0 03 4C 6D D5 A9 00 85
89F0- 0E 4C E9 D5 A0 06 B1 B8
89F8- D9 59 8A D0 0A 88 10 F6
8A00- A2 7C A0 8A 4C 17 8A A0
8A08- 06 B1 B8 D9 60 8A D0 1B
8A10- 88 10 F6 A2 81 A0 8A 20
8A18- B1 00 90 FB 20 7D E0 B0
8A20- F6 A9 00 85 11 85 12 8A
8A28- 4C D8 DE A0 04 B1 B8 D9
8A30- 67 8A D0 1F 88 10 F6 A9
8A38- 71 A0 8A 20 F9 EA A9 86
8A40- A0 8A 20 66 EA A2 97 A0
8A48- 8A 20 2B EB A9 97 A0 8A
8A50- 4C 17 8A 20 E3 DF 4C D8
8A58- DE 54 55 52 54 4C 45 58
8A60- 54 55 52 54 4C 45 59 41
8A68- 4E 47 4C 45 83 49 0F DA
8A70- A1 7B 0E FA 35 12 00 00
```

continued on next page

LISTING 4: TURTLE *(continued)*

```
8A78- 00 00 00 00 00 00 00 00
8A80- 00 00 00 00 00 00 00 00
8A88- 00 00 00 00 00 00 00 00
8A90- 00 00 00 00 00 00 00 00
8A98- 00 00 00 00 00 00 00 00
8AA0- 00 00 00 00 00 00 00 00
8AA8- 00 00 00 00 00 00 00 00
8AB0- 00 00 00 00 00 00 00 00
8AB8- 00 00 A0 00 B1 B8 60 48
8AC0- AD 19 C0 30 FB 68 60 20
8AC8- 67 DD 20 0C E1 A5 A1 A4
8AD0- A0 60 A9 00 8D 76 8A 60
8AD8- A9 FF 8D 76 8A 60 20 67
8AE0- DD A2 92 A0 8A 20 2B EB
8AE8- AD 93 8A 49 80 8D 93 8A
8AF0- 4C FD 8A 20 67 DD A2 92
8AF8- A0 8A 20 2B EB A9 86 A0
8B00- 8A 20 F9 EA 20 EA EF A9
8B08- 92 A0 8A 20 7F E9 A2 A6
8B10- A0 8A 20 2B EB A9 7C A0
8B18- 8A 20 BE E7 A2 97 A0 8A
8B20- 20 2B EB A9 86 A0 8A 20
8B28- F9 EA 20 F1 EF A9 92 A0
8B30- 8A 20 7F E9 A2 AB A0 8A
8B38- 20 2B EB A9 81 A0 8A 20
8B40- BE E7 A2 9C A0 8A 20 2B
8B48- EB 20 D6 8E 60 20 67 DD
8B50- A2 86 A0 8A 20 2B EB A9
8B58- 71 A0 8A 20 7F E9 4C 95
8B60- 8B 20 67 DD A2 97 A0 8A
8B68- 20 2B EB AD 98 8A 49 80
8B70- 8D 98 8A A9 97 A0 8A 20
8B78- F9 EA 4C 87 8B 20 67 DD
8B80- A2 97 A0 8A 20 2B EB A9
8B88- 71 A0 8A 20 7F E9 A9 86
8B90- A0 8A 20 BE E7 A2 97 A0
8B98- 8A 20 2B EB A9 6C A0 8A
8BA0- 20 F9 EA A9 97 A0 8A 20
8BA8- 66 EA 20 23 EC A9 6C A0
8BB0- 8A 20 7F E9 A9 97 A0 8A
8BB8- 20 A7 E7 A2 86 A0 8A 20
8BC0- 2B EB 60 AD 77 8A 29 7F
8BC8- 8D 77 8A 60 AD 77 8A 09
8BD0- 80 8D 77 8A 60 20 BF 8A
8BD8- AD 50 C0 AD 57 C0 60 20
8BE0- BF 8A AD 51 C0 AD 54 C0
8BE8- 60 20 BF 8A AD 53 C0 60
8BF0- 20 BF 8A AD 52 C0 60 20
8BF8- C7 8A 20 BF 8A 4A B0 04
8C00- AD 55 C0 60 AD 54 C0 60
8C08- 20 C7 8A 4A B0 05 A9 40
8C10- 85 E6 60 A9 20 85 E6 60
8C18- 18 A9 1F AA 65 E6 85 27
8C20- A9 00 A0 FF 85 26 2C 76
8C28- 8A 50 01 98 91 26 88 D0
8C30- FB 91 26 C6 27 CA 10 F4
8C38- 60 A2 09 BD 50 8C 9D 7C
8C40- 8A CA 10 F7 A2 03 BD 5A
8C48- 8C 9D 78 8A CA 10 F7 60
8C50- 88 0C 00 00 00 87 40 00
8C58- 00 00 8C 00 60 00 A2 09
8C60- A9 00 9D 7C 8A CA 10 FA
8C68- A2 03 9D 78 8A CA 10 FA
8C70- 60 4C DC D9 A9 03 20 D6
8C78- D3 A5 B9 48 A5 B8 48 A5
8C80- 76 48 A5 75 48 A9 B0 48
8C88- 20 C7 8A 85 50 84 51 20
8C90- 41 D9 4C D2 D7 A2 FF C9
8C98- C1 D0 01 E8 8E 92 8A D0
8CA0- 03 20 B1 00 C9 F2 D0 09
8CA8- 20 B1 00 20 1F 8D 4C D5
8CB0- 8C C9 F3 D0 09 20 B1 00
8CB8- 20 36 8D 4C D5 8C 20 67
8CC0- DD A2 97 A0 8A 20 2B EB
8CC8- 20 BE DE 20 67 DD A2 9C
8CD0- A0 8A 20 2B EB AD 77 8A
8CD8- 48 2C 92 8A 50 08 A9 00
8CE0- 8D 92 8A 20 CC 8B A9 7C
8CE8- A0 8A 20 F9 EA A9 97 A0
8CF0- 8A 20 A7 E7 A2 A6 A0 8A
8CF8- 20 2B EB A9 81 A0 8A 20
8D00- F9 EA A9 9C A0 8A 20 A7
8D08- E7 A2 AB A0 8A 20 2B EB
8D10- 20 D6 8E 68 8D 77 8A 20
8D18- B7 00 C9 C1 F0 83 60 A2
8D20- 09 BD 50 8C 9D 97 8A CA
8D28- 10 F7 A2 03 BD 5A 8C 9D
8D30- 78 8A CA 10 F7 60 A2 09
8D38- A9 00 9D 97 8A CA 10 FA
8D40- A2 03 9D 78 8A CA 10 FA
8D48- 60 20 BA 8A C9 28 D0 3B
8D50- 20 B1 00 20 67 DD A2 97
8D58- A0 8A 20 2B EB 20 BE DE
8D60- 20 67 DD A2 9C A0 8A 20
8D68- 2B EB 20 B8 DE 20 AE 8D
8D70- AD B8 8A 8D 8F 8A A2 FF
8D78- AD B6 8A E8 C9 07 90 05
8D80- E9 07 4C 7B 8D 8E 8D 8A
8D88- 8D 8E 8A A5 36 8D 90 8A
8D90- A5 37 8D 91 8A A9 F6 85
8D98- 36 A9 8D 85 37 20 B7 00
8DA0- 20 D5 DA AD 90 8A 85 36
8DA8- AD 91 8A 85 37 60 A9 97
8DB0- A0 8A 20 F9 EA 20 0C E1
8DB8- A5 A1 8D B6 8A A5 A0 8D
8DC0- B7 8A A9 9C A0 8A 20 F9
8DC8- EA 20 0C E1 A5 A1 8D B8
8DD0- 8A A5 A0 8D B9 8A AD B7
8DD8- 8A F0 0B C9 02 B0 14 AD
8DE0- B6 8A C9 18 B0 0D AD B9
8DE8- 8A D0 08 AD B8 8A C9 C0
8DF0- B0 01 60 4C 06 F2 85 D0
8DF8- 48 8A 48 98 48 A5 D0 29
8E00- 7F C9 20 B0 14 C9 0D D0
8E08- 06 20 61 8E 4C 5B 8E C9
8E10- 07 D0 06 20 DD FB 4C 5B
8E18- 8E E9 20 85 D0 A9 00 06
8E20- D0 2A 06 D0 2A 06 D0 2A
8E28- 85 D1 A5 D0 6D 8B 8A 85
8E30- D0 A5 D1 6D 8C 8A 85 D1
8E38- AD 8D 8A 8D B2 8A A0 07
8E40- 8C B8 8A 20 76 8E 20 AD
8E48- 8E CE B8 8A 10 F5 EE 8D
8E50- 8A AD 8D 8A C9 28 90 03
8E58- 20 61 8E 68 A8 68 AA 68
8E60- 60 A9 00 8D 8D 8A 18 AD
8E68- 8F 8A 69 08 C9 B9 90 02
8E70- A9 00 8D 8F 8A 60 18 AD
8E78- 8F 8A 6D B8 8A A2 00 86
8E80- 27 AA 29 38 0A 0A 0A 26
8E88- 27 0A 26 27 85 26 8A 29
8E90- C0 4A 8D B3 8A 4A 4A 0D
8E98- B3 8A 05 26 6D B2 8A 85
8EA0- 26 8A 29 07 0A 0A 05 27
8EA8- 65 E6 85 27 60 AC B8 8A
8EB0- B1 D0 A0 00 AE 8E 8A F0
8EB8- 19 84 9D 0A 26 9D CA D0
8EC0- FA 0A 26 9D 4A 20 AE 90
8EC8- AE 8D 8A E0 27 B0 06 C8
8ED0- A5 9D 20 AE 90 60 A9 7C
8ED8- A0 8A 20 F9 EA 20 A0 E7
8EE0- 20 0C E1 38 A5 A1 ED 78
8EE8- 8A 10 05 49 FF 18 69 01
8EF0- C9 02 90 0A A5 A0 8D 79
8EF8- 8A A5 A1 8D 78 8A A9 81
8F00- A0 8A 20 F9 EA 20 A0 E7
8F08- 20 0C E1 38 A5 A1 ED 7A
8F10- 8A 10 05 49 FF 18 69 01
8F18- C9 02 90 0A A5 A0 8D 7B
8F20- 8A A5 A1 8D 7A 8A A9 97
8F28- A0 8A 20 F9 EA 20 0C E1
```

continued on page 36

NORTHEASTERN Software

88 Ryders Lane, Stratford, Connecticut 06497

SPECIALS

Appleworks	174.00
Back It Up III	42.00
CIA Files	46.75
Copy II+	19.00
Copy Master II 128K w/Gr. Dump	94.00
EDD III	43.50
Fokker Triplane	35.00
Spellworks	37.00
Jeevz	35.00
Locksmith 5.0	64.00
R. Wagner Mouse Write	72.00
Super Serial Card	104.00
Wildcard II	70.00
Pinpoint	CALL

ENTERTAINMENT

Spy Hunter or Buck Rogers	26.00
Tapper or Star Trek	26.00
Frogger 3-Deep or Super Zaxxon	21.75
Sorcerer, Suspect or Infidel	26.00
Cutthroats or Zork I	22.00
Zork II or Zork III	25.00
Witness or Seastalker	22.00
Deadline, Suspended, or Starcross	28.00
Planetfall or Enchanter	22.00
Hitchhiker's Guide to the Galaxy	22.00
Murder by the Dozen or Felony	22.00
Ultima III	33.50
Lode Runner or Karateka	20.00
Championship Lode Runner	20.00
Choplifter	20.00
Reforger 88 or Kampfgruppe	35.50
50 Mission Crush or Rails West	23.50
Objective Kursk or Phantasie	23.50
Questron	29.75
Battle for Normandy	23.75
Operation Market Garden	29.75
Imperium Galacticum or Broadsides	23.50
Geopolitique 1990	23.50
Baltic 85 or Gemstone Warrior	20.50
North Atlantic 86	35.00
Germany 1985 or Carrier Force	35.00
War in Russia	47.00
Flight Simulator II	30.75
Night Mission Pinball	21.00
Wizardry I	29.00
Wizardry II	20.75
Wizardry III or Wizardry IV	23.75
Wizprint	14.75
Rendezvous with Rama	23.00
Dragonworld or Shadowkeep	23.00
F-15 Strike Eagle or Solo Flight	20.75
NATO Commander	21.75
Crusade in Europe	24.00
Fahrenheit 451	23.00
Amazon	23.00
Expedition Amazon	20.00
Pole Position or Zaxxon	18.75
The Tracer Sanction or H.E.R.O.	23.75
Pitfall II or Rock N Bolt	23.50
Ghostbusters or Zenii	23.50
Mind Shadow or Space Shuttle	23.50
The Quest or Ring Quest	20.50
Coveted Mirror	20.75
Xyphus or Transylvania	20.50
Frogger or BC's Quest for Tires	20.75
King's Quest	29.00
King's Quest II	29.00
Micro League Baseball	23.75
Where...Is Carmen San Diego?	23.75
Summer Games I or II	23.75
Ken Uston's Blackjack	41.00
Castle Wolfenstein	19.75
Beyond Castle Wolfenstein	19.50
Millionaire, Baron, or Tycoon	23.75
Forbidden Quest	23.75
Spitfire Simulator	24.75
AirSim III	25.75
Bruce Lee or Conan	22.50
Sargon III	27.50
Death in the Caribbean	20.75
Miner 2049er or The Heist	23.75
How About A Nice Game of Chess?	20.75
Mask of the Sun	26.75
Adventure Construction Set	30.75
Europe Ablaze	30.75
Carriers at War	30.75
Bard's Tale or Reach for The Stars	28.50
Hard Hat Mack	12.75
Pinball Construction Set	21.50
Music Construction Set	24.75
The Standing Stone	12.75
One-On-One	24.50
Archon	22.25
Archon II or Skyfox	24.50
Seven Cities of Gold	24.50
Murder On The Zinderneuf	13.50
Sundog	22.75
Gato	22.75

PRINTERS

APPLE	
Imagewriter II (NEW!)	445.00
Imagewriter 15", UNDER	399.00
BROTHER	
HR-15 XL	469.00
HR-35	354.00
C. ITOH	
8510AP	340.00
8510AP Plus	285.00
1550AP	315.00
1550AP Plus	430.00
DYNAX	
DX-15	450.00
EPSON	
Spectrum LX-80	209.00
Spectrum LX-90	CALL
JX-80	CALL
FX-85 or FX-185	CALL
LQ-1500	CALL
DX-10 or DX-20	CALL
JUKI	
6100***	347.00
6300	652.00
MANNESMAN TALLY	
MT-85	340.00
MT-86	410.00
MT 290	610.00
MT 490	1570.00
MT 460	1340.00
NEC INFO SYSTEMS	
N2030	615.00
N3530	955.00
OKIDATA	
182	210.00
192	330.00
192 Imagewriter	360.00
193	470.00
193 Imagewriter	53.00
PANASONIC	
KXP 1091***	235.00
KXP 1092	359.00
KXP 1093	420.00
KXP 3151	395.00
QUME	
Qume Printer Line	CALL
SILVER REED	
EXP 400P	200.00
EXP 500P	235.00
EXP 550P	369.00
EXP 770P	669.00
STAR MICRONICS	
SG 10	219.00
SG 15	369.00
SR 10	479.00
SR 15	579.00
SD 10	329.00
SD 15	439.00
TOSHIBA	
P1340	435.00
P351	1015.00
P341	885.00

ACCESSORIES/CARDS/ HOME SOFTWARE

Accelerator IIE (TITAN)	204.00
Apple Mechanic	16.00
Apple Mouse IIc	74.00
Apple Mouse IIe	104.00
Applesoft Compiler	104.00
Appleworks Modifier 64K	38.00
Appleworks Modifier 128K	48.00
Applied Eng. 0K to 1MB	CALL
Applied Eng. System Clock	63.00
Applied Eng. Viewmaster	103.00
Applied Eng. Z-80 Plus	104.00
Apricorn 16K RAM Card	65.00
Apricorn 64K Extend It	85.00
ASCII Pro	75.00
ASC TMulti I/O	155.00
Bag of Tricks (QUALITY)	24.00
Bank St. Filer, Speller, or Writer	40.00
Beagle Basic	19.00
Beagle Graphics	31.00
Blazing Paddles	27.00
Buffered Grappler	133.00
Business Card (STREET ELEC.)	131.00
Checkmate Multiview 80/160	215.00
Compuserve Starter Kit	CALL
Cricket IIc	105.00
Crossword Magic	30.00
Curtis Surge Protectors, From	28.00
Cut & Paste	23.00
D-Code	22.00
Dazzle Draw	35.00
Digital Paintbrush System	174.00
Disk Quick	16.00
Dollars & Sense	57.00
Dollars & Sense IIC	67.00
Doss Boss	13.00
Double Take	19.00
Echo II Plus	92.00
Extend 80 (CARIBBEAN)	78.00
Extra K	22.00
Fat Cat	19.00
Financial Cookbook	31.00
Fingerprint Plus (THIRDWARE)	95.00
Flex Text	16.00
Flip N File 50	18.00
Font Paks 1-11 (EACH)	11.25
Fontrix 1.5	54.00
Forecast	36.00
Frame Up	16.50
GPLE	26.70
Gibson Light Pen	144.00
Graphics Library 1 or 2	14.50
Graphics Magician, The	34.00
Grappler Plus	73.00
Hayes Mach II	26.50
Hayes Mach III	32.50
Hayes Paddles	26.50
Head Cleaning Kit	20.00
I/O Silver	16.75
Keytronics Keyboards	186.00
Koala Pad	71.00
Kraft Joystick	26.00
Lazerware Lisa V2.6	53.00
MCT Speed Demon	158.00
Merlin	34.00
Merlin Combo	65.00
Merlin Pro	56.00
Micro Cookbook II/IIE	22.25
Mockingboard A or B	67.00
Mockingboard C	119.00
Mockingboard D	129.00
Money Street	74.00
Muppet Learning Keys	42.00
Neptune 80 Col. 64K Card	183.00
Net Worth	47.00
New Complete Graphics System	46.00
Nice Print	105.00
Pixit	28.75
Premium Softcard IIe	248.00
Print It (TEXTPRINT)	128.00
Print Shop	27.00
Pro Byter	19.25
Proto Doss	16.25
Quadram Multi-Core Board	CALL
Quadram E-Ram 80	98.00
Saturn 128K Ramcard	174.00
SeriAll Interface Card	91.00
Sideways	33.00
Silicon Salad	13.50
Softerm II (SOFTRONICS)	123.00
System Saver (KENSINGTON)	58.00
Take One	34.00
TG Selectaport	24.75
Thunderclock Plus	104.00
Time Is Money (TURNING POINT)	58.00
Time Trax IIE or IIC	78.00
Trackhouse Keypads	CALL
Triple Dump	21.50
Turbo Pascal	35.95
Turbo Toolbox	29.95
Turbo Tutor	20.75
Type Faces	10.95
Typing Tutor II	14.90
Typing Tutor III	29.75
Ultraterm w/SS Inv.	176.00
Utility City	16.25
Video 7 RGB Interface IIE	94.00
Video 7 RGB Interface IIC	85.00
Visible 6502	31.75
Zoom Graphics	30.00

MONITORS

APPLE	
Color Monitor IIe or IIc	280.00
AMDEK	
Color 300***	209.00
Color 500	245.00
Color 600	375.00
Video 300A Amber***	118.00
Video 300G Green***	112.00
DVM III	
PRINCETON	
HX-9	399.00
HX-12	419.00
RBG Interface	147.00
TAXAN	
115 Green	110.00
116 Amber	115.00
220 Composite Color	220.00
420 Hi Res RGB Color	375.00
610	320.00
620	370.00
630	410.00
640	465.00

BLANK MEDIA

Elephant SS/DD	12.00
Elephant DS/DD	15.50
Verbatim SS/DD	14.50
Verbatim DS/DD	18.50
Memorex SS/DD	14.00
BASF SS/DD	11.00
Maxell SS/DD	14.00

EDUCATIONAL

APPLE	
Logo II	77.00
BARRONS	
Computer SAT or GMAT	52.00
CBS	
Adventure Master	25.50
Mastering the SAT	57.00
Forecast or T-Rex	27.50
DAVIDSON AND ASSOCIATES	
Speed Reader II	38.50
Word Attack, Mathblaster, or Spellit	27.50
All Davidson Data Disks	11.95
DESIGNWARE	
Spellcopter or Spellakazam	24.00
States & Traits or European Nations	26.00
Body Transparent or Mission Algebra	26.00
Remember	44.00
DLM	
Alien Addition or Alligator Mix	19.00
Meteor Mult. or Minus Mission	19.00
Demolition Division or Dragon Mix	19.00
Spelling Wiz or Word Invasion	19.00
EDU-WARE/PEACHTREE	
Compu Read	16.00
Compu Math	26.50
SAT or PSAT Word Attack Skills	16.00
Spelling Bee & Reading Primer	22.00
Algebra 1, 2, 3, or 4	21.00
Algebra 5 & 6	25.75
Writing Skills 1, 2, 3, 4, or 5	22.00
Hands on BASIC	25.50
Introduction to Counting	19.50
Fractions or Decimals	19.50
HARCOURT BRACE JOVANOVICH	
Harcourt SAT	45.50
Harcourt GRE or ACT	52.50
GROLIER	
Science Island or Treasure Hunter	24.00
Easy Graph or Friendly Filer	24.00
HAYDEN	
Word Challenge	12.00
LEARNING COMPANY	
Rocky's Boots or Robot Odyssey	28.00
Number Stumper or Reader Rabbit	23.00
Magic Spells or Word Spinner	20.75
Bumble Games or Bumble Plot	23.75
Gertrude's Puzzle or Secrets	25.00
MAGNUM	
Super Speed Reading	30.00
MECC	
Elementary Series (EACH)	29.00
Math or Health Series (EACH)	29.00
MILLIKEN	
Math Sequences (EACH)	21.00
Number Chase or Pop R' Spell	24.00
PROGRAM DESIGN	
The New Step By Step	52.00
Step By Step II or III	52.00
SCARBOROUGH SYSTEMS	
Build A Book About You	23.50
Boston Computer Diet	46.00
Mastertype's Writing Wizard	46.00
Mastertype's Filer	23.50
Mastertype's New & Improved	23.50
Run for the Money or Songwriter	23.50
SCHOLASTIC WIZWARE	
Fact and Fiction Toolkit IIC	23.75
Microzine 1-8 (EACH)	23.75
Story Tree or Story Maker	23.75
Turtle Tracks or Agent USA	23.75
Operation Frog or Logic Builders	23.75
Logo Robot or Kids At Work	15.00
SPINNAKER SOFTWARE	
Fraction Fever or Face Maker	14.75
File and Report or Spreadsheet	34.00
Snooper Troops 1 or 2	19.75
Story Machine or Grandma's House	14.75
Kindercomp or Kids on Keys	14.75
Word Processor	34.75
Alphabet Zoo or Kidwriter	16.75
In Search Of	19.75
SPRINGBOARD	
Early Games or Piece of Cake	16.00
Make A Match or Fraction Factory	16.00
The Newsroom	28.00
Clip Art Effects for Newsroom	16.00
SUNBURST	
Challenge Math or The Factory	29.75
Teaser by Tobbs	29.75
The Incredible Laboratory	29.75
TERRAPIN	
Terrapin Logo	55.00
Terrapin Logo 10 Pack	178.00
TIMEWORKS	
Evelyn Wood Dynamic Reader	41.75
WEEKLY READER	
Sticky Bear Series, (EACH)	23.75

BUSINESS

ADVANCED LOGIC SYSTEMS	
The Handlers Package	95.00
List or Word Handler	44.00
APPLE	
Apple Writer IIE	112.00
Apple Access	54.00
Quickfile	68.00
APPLIED SOFTWARE TECHNOLOGY	
Versaform	44.00
ARTSCI	
Magic Office Systems	123.00
Magic Calc 3.3	55.50
Magic Window II	76.50
ASHTON TATE	
DBase II	250.50
BPI	
Gen. Acctg., AR, AP, GL, PAY	195.00
Gen. Acctg., AR, AP, GL, PAY Pro Dos	210.00
BLUE CHIP SOFTWARE	
Omnis III	227.00
CONTINENTAL SOFTWARE	
FCM/FL	57.00
Home Accountant or Tax Advantage	38.00
GL, AR, AP, PAY	133.00
DOW JONES SOFTWARE	
Investor's Workshop	86.00
Market Analyzer or Microscope	198.00
Mkt. Mgr. or Spreadsheet Link	147.00
Membership Kit	18.00
HAYDEN SOFTWARE	
The Writer	27.00
HOWARD SOFTWARE SERVICES	
Real Estate Analyzer III	CALL
HUMAN EDGE	
Communication or Sales Edge	88.00
Mgt. or Negotiation Edge	88.00
Mind Prober	26.00
KENSINGTON	
Format II Enhanced	86.50
LIVING VIDEOTEXT	
Think Tank II/IIE	77.00
MEGAHAUS	
Megafinder	89.00
Megaworks	66.50
MICROPRO	
Wordstar	168.00
Wordstar Professional	227.00
MICROSOFT	
Multiplan	54.00
MICROSTUF	
Crosstalk	103.00
PEACHTREE	
Back To Basics Accounting	99.00
PRACTICORP	
Practicalc	28.00
PRECISION SOFTWARE	
Superbase	54.00
QUARK	
Word Juggler w/Lexicheck	59.00
SENSIBLE SOFTWARE	
Bookends or Sensible Speller	67.00
The Graphics Department	65.00
Report Card	33.00
SIERRA-ON-LINE	
The General Manager II	127.00
Homeword	41.00
Homeword Speller	30.00
Screenwriter IIE	73.00
SORCIM	
Supercalc 3A	101.00
SOFTWARE PUBLISHING	
PFS: Best Sellers	142.00
PFS: Access or Proof	44.00
PFS: File or Graph	68.50
PFS: Report or Write	68.50
STONEWARE	
DB Master 4 Plus	167.00
DB Master Hard Disk	235.00
DB Master Business Writer	87.00
DB Bus. Pack or Master Bus. Team	187.00
VISICORP	
Flashcalc	57.00

DISK DRIVES

APPLE	
UNIDISK w/C	255.00
Duo Disk w/Acc. Kit	425.00
IIC Add-On	219.00
MICRO SCI	
A2 143K	152.00
Slimline IIE	152.00
Slimline IIC	159.00
Controller Card	47.00
QUARK	
QC-10 Hard Disk, UNDER	999.00
QC-20 Hard Disk, UNDER	1379.00

MODEMS

212 Apple Cat	353.00
Apple Cat II	187.00
Apple Personal Modem	260.00
Hayes Smartmodem 300IIc	152.00
Hayes Micromodem IIE w/SC I	128.00
Hayes Smartmodem 1200	360.00
Popcom X-100 Modem	259.00
Prometheus 1200 External	290.00
Prometheus Internal IIE/II+	263.00
Zoom Modem IIE	114.00

*** Top rated in a leading consumer magazine

WE WILL BEAT ANY COMPARABLE ADVERTISED PRICE BY $1 (See Below)
SCHOOL AND CORPORATE PURCHASE ORDERS ACCEPTED; CALL 1-800-874-1108

MasterCard VISA

Personal and company checks allow 3 weeks to clear. For faster delivery, send cashier's check, certified check, or money order. Shipping—Software ($3.00 minimum). Shipping—Hardware (please call 203-375-3860). COD—Add an additional $2.00. Alaska, Hawaii, Canada, PO, APO, and FPO $5.00 minimum. Foreign orders—$15.00 minimum and 15% of all orders over $100. Mastercard and Visa (please include card no. and expiration date). Connecticut residents add 7.5% sales tax. Prices subject to change without notice. All returns must have a return authorization number. Call 203-375-3860 to obtain one before returning goods for replacement. Defective merchandise replaced with same item only. We do not guarantee compatibility. All sales are final.

"Comparable advertised price" does not include typographical errors on the part of other vendors or prices that do not reflect higher shipping costs, credit card charges, or membership fees. The $1 off offer only applies to items where Northeastern's price is not already the lowest. Final judgement remains with the manager on duty.

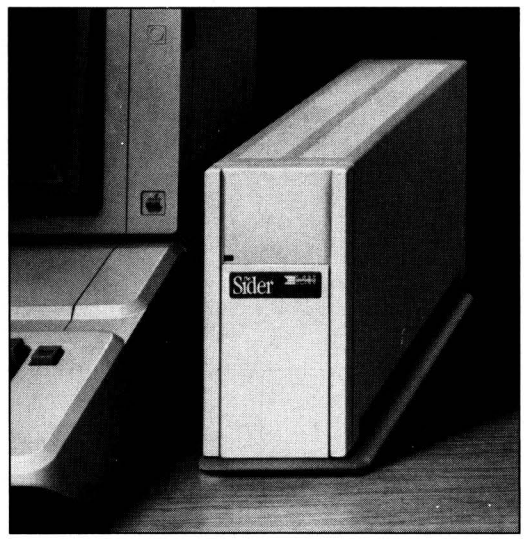
LISTING 4: TURTLE *(continued)*

```
8F30- A5 A1 8D B2 8A A5 A0 8D
8F38- B3 8A A9 9C A0 8A 20 F9
8F40- EA 20 0C E1 A5 A1 8D B4
8F48- 8A A5 A0 8D B5 8A A2 04
8F50- BD 97 8A 9D 7C 8A BD 9C
8F58- 8A 9D 81 8A CA 10 F1 20
8F60- 6D 8F A2 03 B5 A5 9D 78
8F68- 8A CA 10 F8 60 A9 00 8D
8F70- B2 8A AD A7 8A 8D B0 8A
8F78- AD AC 8A 8D B1 8A AD A7
8F80- 8A 29 7F 8D A7 8A AD AC
8F88- 8A 29 7F 8D AC 8A A2 00
8F90- BD A6 8A DD AB 8A D0 05
8F98- E8 E0 05 90 F3 B0 04 20
8FA0- 4C 90 60 20 09 90 60 A5
8FA8- A8 10 05 A9 00 4C 7D 8E
8FB0- F0 05 A9 BF 4C 7D 8E A5
8FB8- A7 C9 C0 90 02 A9 BF 4C
8FC0- 7D 8E A5 A5 85 A2 A5 A6
8FC8- 85 A3 10 09 A9 00 85 A2
8FD0- 85 A3 4C E8 8F 38 A5 A2
8FD8- E9 17 A5 A3 E9 01 90 08
8FE0- A9 17 85 A2 A9 01 85 A3
8FE8- 06 A2 26 A3 A0 07 A9 00
8FF0- 85 9E 88 30 11 26 9E 06
8FF8- A2 26 A3 38 A5 A3 E9 07
9000- 90 F0 85 A3 B0 EC 26 9E
9008- 60 AD A6 8A D0 0B A2 04
9010- 9D A1 8A CA 10 FA 4C 2E
9018- 90 A9 A6 A0 8A 20 F9 EA
9020- A9 AB A0 8A 20 66 EA A2
9028- A1 A0 8A 20 2B EB A9 A6
9030- A0 8A 20 3F 91 20 8F 90
9038- 20 BF 90 D0 01 60 20 B4
9040- 91 90 03 20 C6 90 20 E5
9048- 90 4C 35 90 AD AB 8A D0
9050- 0B A2 04 9D A1 8A CA 10
9058- FA 4C 71 90 A9 AB A0 8A
9060- 20 F9 EA A9 A6 A0 8A 20
9068- 66 EA A2 A1 A0 8A 20 2B
9070- EB A9 AB A0 8A 20 3F 91
9078- 20 8F 90 20 BF 90 D0 01
9080- 60 20 B4 91 90 03 20 E5
9088- 90 20 C6 90 4C 78 90 AD
9090- 77 8A D0 2A A5 A8 D0 26
9098- A5 A7 C9 C0 B0 20 A5 A5
90A0- E9 17 A5 A6 30 18 E9 01
90A8- B0 14 A4 9E A5 9D 2C 76
90B0- 8A 70 05 11 26 91 26 60
90B8- 49 7F 31 26 91 26 60 E6
90C0- A1 D0 02 E6 A0 60 AD B1
90C8- 8A 10 10 38 A5 A7 E9 01
90D0- 85 A7 A5 A8 E9 00 85 A8
90D8- 4C E1 90 E6 A7 D0 02 E6
90E0- A8 20 A7 8F 60 AD B0 8A
90E8- 10 2D 38 A5 A5 E9 01 85
90F0- A5 A5 A6 E9 00 85 A6 38
90F8- A5 A5 E9 17 A5 A6 E9 01
9100- B0 14 46 9D 90 10 A9 40
9108- 85 9D C6 9E 10 08 A9 00
9110- 85 9E A9 01 85 9D 60 E6
9118- A5 D0 02 E6 A6 A5 A6 30
9120- 18 06 9D 10 14 A9 01 85
9128- 9D E6 9E A5 9E C9 28 30
9130- 08 A9 27 85 9E A9 40 85
9138- 9D 60 80 40 00 00 00 20
9140- F9 EA A9 3A A0 91 20 BE
9148- E7 20 0C E1 A5 A0 49 FF
9150- 85 A0 A5 A1 49 FF 85 A1
9158- A2 03 BD 78 8A 95 A5 CA
9160- 10 F8 AD A2 8A 09 80 85
9168- AA 85 AC AD A3 8A 85 A9
9170- 85 AB AC A1 8A C0 81 D0
9178- 0D A9 FF 85 A9 85 AA 85
9180- AB 85 AC 4C 9F 91 C0 71
9188- B0 09 A9 00 85 A9 85 AA
9190- 4C 9F 91 C0 80 B0 08 46
9198- AA 66 A9 C8 4C 93 91 20
91A0- A7 8F 20 C2 8F A4 A3 B9
91A8- AD 91 85 9D 60 01 02 04
91B0- 08 10 20 40 18 A5 AB 65
91B8- A9 85 AB A5 AC 65 AA 85
91C0- AC 60 00 00 00 00 00 00
91C8- 00 00 00 00 00 00 00 00
91D0- 08 08 08 08 00 00 08 00
91D8- 14 14 14 00 00 00 00 00
91E0- 14 14 3E 14 3E 14 14 00
91E8- 08 3C 0A 1C 28 1E 08 00
91F0- 06 26 10 08 04 32 30 00
91F8- 04 0A 0A 04 2A 12 2C 00
9200- 08 08 08 00 00 00 00 00
9208- 08 04 02 02 02 04 08 00
9210- 08 10 20 20 20 10 08 00
9218- 08 2A 1C 08 1C 2A 08 00
9220- 00 08 08 3E 08 08 00 00
9228- 00 00 00 00 08 08 04 00
9230- 00 00 00 3E 00 00 00 00
9238- 00 00 00 00 00 00 08 00
9240- 00 20 10 08 04 02 00 00
9248- 1C 22 32 2A 26 22 1C 00
9250- 08 0C 08 08 08 08 1C 00
9258- 1C 22 20 18 04 02 3E 00
9260- 3E 20 10 18 20 22 1C 00
9268- 10 18 14 12 3E 10 10 00
9270- 3E 02 1E 20 20 22 1C 00
9278- 38 04 02 1E 22 22 1C 00
9280- 3E 20 10 08 04 04 04 00
9288- 1C 22 22 1C 22 22 1C 00
9290- 1C 22 22 3C 20 10 0E 00
9298- 00 00 08 00 08 00 00 00
92A0- 00 00 08 00 08 08 04 00
92A8- 10 08 04 02 04 08 10 00
92B0- 00 00 3E 00 3E 00 00 00
92B8- 04 08 10 20 10 08 04 00
92C0- 1C 22 10 08 08 00 08 00
92C8- 1C 22 2A 3A 1A 02 3C 00
92D0- 08 14 22 22 3E 22 22 00
92D8- 1E 22 22 1E 22 22 1E 00
92E0- 1C 22 02 02 02 22 1C 00
92E8- 1E 22 22 22 22 22 1E 00
92F0- 3E 02 02 1E 02 02 3E 00
92F8- 3E 02 02 1E 02 02 02 00
9300- 3C 02 02 02 32 22 3C 00
9308- 22 22 22 3E 22 22 22 00
9310- 1C 08 08 08 08 08 1C 00
9318- 20 20 20 20 20 22 1C 00
9320- 22 12 0A 06 0A 12 22 00
9328- 02 02 02 02 02 02 3E 00
9330- 22 36 2A 2A 22 22 22 00
9338- 22 22 26 2A 32 22 22 00
9340- 1C 22 22 22 22 22 1C 00
9348- 1E 22 22 1E 02 02 02 00
9350- 1C 22 22 22 2A 12 2C 00
9358- 1E 22 22 1E 0A 12 22 00
9360- 1C 22 02 1C 20 22 1C 00
9368- 3E 08 08 08 08 08 08 00
9370- 22 22 22 22 22 22 1C 00
9378- 22 22 22 22 14 14 08 00
9380- 22 22 22 2A 2A 36 22 00
9388- 22 22 14 08 14 22 22 00
9390- 22 22 14 08 08 08 08 00
9398- 3E 20 10 08 04 02 3E 00
93A0- 3C 04 04 04 04 04 3C 00
93A8- 00 02 04 08 10 20 00 00
93B0- 1E 10 10 10 10 10 1E 00
93B8- 08 1C 2A 08 08 08 00 00
93C0- 00 00 00 00 00 00 3E 00
93C8- 00 00 00 00 00 00 00 00
93D0- 00 00 0E 10 1C 12 2C 00
93D8- 02 02 1A 26 22 22 1E 00
93E0- 00 00 1C 22 02 02 3C 00
```

continued on page 38

What every Apple II+ and IIe user should ask before buying the "Sider" 10 MB hard disk:

When a company offers a superior quality 10 megabyte Winchester hard disk for only $695, it's bound to raise a few eyebrows...and a lot of questions. The fact is, you're probably already wondering "Can I really get a 10 megabyte hard disk that's *reliable* for only $695?" The answer is: ABSOLUTELY...when you choose the Sider from First Class Peripherals.



What's so great about the Sider?

For starters, the Sider lets you boot your ***Apple II+*** or ***IIe*** directly off the hard disk—unlike some other Winchester subsystems. Rebooting is also trouble-free. And the disk is partitionable, allowing you to allocate space to four operating systems on the same disk. The Sider supports: Apple DOS 3.3; Pro DOS™; Apple Pascal; and CP/M®

What's more, a small "footprint" lets you incorporate the compact Sider into your existing computer set-up with ease.

In addition, with the Sider, you not only pay far less for the subsystem, you also save money on installation. Because, unlike other 10 MB systems that require the purchase of expensive "extras," the Sider is *plug and play*. Everything you need is provided, including cables, host adaptor, installation software and manual.

What makes it so reliable?

To start, the Sider is manufactured, and sold exclusively, by First Class Peripherals, an innovative computer company which is backed by Xebec. The computer industry's leading manufacturer of disk controllers, Xebec has over a decade of experience serving customers like IBM, Toshiba, Texas Instruments and Hewlett Packard. It's this kind of expertise that helps assure the Sider's performance.

Special design features further enhance reliability. The Sider's controller is the field-proven, industry standard Xebec S1410A. And Xebec's 3200 drive tester, the toughest in the industry, ensures that the Sider will operate reliably. One more assurance of the $695 Sider's quality: it's UL Approved and FCC Class B rated.

"Only $595"

OCTOBER 1 thru DECEMBER 31, 1985

But why is it only $695?

You pay less for the Sider than for other 10 MB hard disks simply because you're paying for the superior quality components inside the unit, not for a lot of retail overhead costs. Since First Class Peripherals sells direct, you avoid dealer and distribution expenses, and pay only for the product.

What about a guarantee?

Like many experienced Apple users, you may be reluctant to buy a hard disk priced at only $695 without first seeing for yourself how it performs. That's why First Class Peripherals offers you a reassuring, money-back guarantee that eliminates any risk on your part. Simply order the Sider and use it for 15 days. Then, if you're not entirely satisfied, return it and receive a full refund—no questions asked.

The Sider also comes with a full one-year limited warranty. Plus, there's a convenient, toll-free hotline you can call anytime you have a technical or service question, or need help.

Don't delay. Order the Sider now.

To receive the Sider 10 megabyte Winchester hard disk subsystem for only $695, simply order using the coupon below. For faster service, order by phone and charge to your VISA, MasterCard or American Express. (You can also call us if you have any questions or technical concerns about the Sider. We'll see you get the help you need.) Call toll-free:

U.S.: 1-800-538-1307, Ext. 204
CANADA: 1-800-227-7792, Ext. 204

☐ Yes, please send me the Sider, including half-height 10 megabyte Winchester hard disk drive, Apple adaptor board, cable, complete installation software and documentation.

I prefer to pay as follows:

☐ I've enclosed my check or money order for $695* + $15 shipping and handling, payable to First Class Peripherals.

☐ Please bill the following credit card account for $695* + $15 shipping and handling:

☐ VISA ☐ MasterCard ☐ American Express

Card # *Exp. Date*

Signature

Name

Address

City

State *Zip*

Telephone (area code)

Mail to:

FIRST PERIPHERALS CLASS

3579 Highway 50 East
Carson City, NV 89701

*Residents of CA, NV and PA, please add appropriate sales tax.

Apple and Pro DOS are trademarks of Apple Computer, Inc.
CP/M is a registered trademark of Digital Research, Inc.

LISTING 4: TURTLE *(continued)*

```
93E8- 20 20 2C 32 22 22 3C 00
93F0- 00 00 1C 22 3E 02 3C 00
93F8- 18 24 04 0E 04 04 04 00
9400- 00 00 3C 22 22 3C 20 1E
9408- 02 02 1A 26 22 22 22 00
9410- 08 00 0C 08 08 08 1C 00
9418- 10 00 10 10 10 10 12 0C
9420- 02 02 22 12 0A 16 22 00
9428- 0C 08 08 08 08 08 1C 00
9430- 00 00 16 2A 2A 2A 2A 00
9438- 00 00 1A 24 24 24 24 00
9440- 00 00 1C 22 22 22 1C 00
9448- 00 00 1E 22 22 1E 02 02
9450- 00 00 3C 22 22 3C 20 20
9458- 00 00 1A 26 02 02 02 00
9460- 00 00 3C 02 1C 20 1E 00
9468- 00 04 0E 04 04 04 18 00
9470- 00 00 12 12 12 12 2C 00
9478- 00 00 22 22 22 14 08 00
9480- 00 00 22 22 2A 2A 14 00
9488- 00 00 22 14 08 14 22 00
9490- 00 00 22 22 22 3C 20 1E
9498- 00 00 3E 10 08 04 3E 00
94A0- 10 08 08 04 08 08 10 00
94A8- 00 08 08 08 08 08 08 00
94B0- 04 08 08 10 08 08 04 00
94B8- 00 00 04 2A 10 00 00 00
94C0- 7F 7F 7F 7F 7F 7F 7F 7F
```

END OF LISTING 4

```
          KEY PERFECT 5.0
              RUN ON
              TURTLE
=====================================
CODE-5.0   ADDR# - ADDR#   CODE-4.0
--------   ------------    --------
A111E0E5    8800 - 884F        299A
E8A49E43    8850 - 889F        2B01
F1783BB6    88A0 - 88EF        23DC
32EEDF73    88F0 - 893F        280C
7CF6D448    8940 - 898F        2799
3E19F92E    8990 - 89DF        2643
27B1C53E    89E0 - 8A2F        2D34
8001C39C    8A30 - 8A7F        2B2F
47BC6D84    8A80 - 8ACF        0AC4
314905CE    8AD0 - 8B1F        248F
CE5C46D1    8B20 - 8B6F        288F
0517E253    8B70 - 8BBF        2226
E7989E91    8BC0 - 8C0F        254F
E970252A    8C10 - 8C5F        28B1
E7F9EE6B    8C60 - 8CAF        276C
78411D8B    8CB0 - 8CFF        2455
761D263E    8D00 - 8D4F        2B6A
8637968B    8D50 - 8D9F        26B1
44E012F2    8DA0 - 8DEF        2929
4BC231D0    8DF0 - 8E3F        2640
21C7CCD4    8E40 - 8E8F        218D
B264882E    8E90 - 8EDF        29D1
A698E43E    8EE0 - 8F2F        2E6B
339B1124    8F30 - 8F7F        280D
CB032027    8F80 - 8FCF        2273
406D791C    8FD0 - 901F        2767
86DF9D24    9020 - 906F        27A0
4F1003F5    9070 - 90BF        2E05
B6836058    90C0 - 910F        2935
17DCA5F4    9110 - 915F        23CD
D192B28D    9160 - 91AF        2894
8C12AFAD    91B0 - 91FF        2741
FFA12704    9200 - 924F        2A2C
6F6A3532    9250 - 929F        24BF
4CFCD37F    92A0 - 92EF        2884
789D0584    92F0 - 933F        2981
37B63830    9340 - 938F        2A54
8C74F253    9390 - 93DF        26DE
23C119C7    93E0 - 942F        29E4
512869CB    9430 - 947F        23B0
40F1C498    9480 - 94C7        2318
325BB538 = PROGRAM TOTAL =     0CC8
```

LISTING 5: DEMO1

```
1   REM ***********************
2   REM *        DEMO1          *
3   REM *     BY J.B. WARD      *
4   REM * COPYRIGHT (C) 1986 *
5   REM * BY MICROSPARC, INC *
6   REM * CONCORD, MA  01742 *
7   REM ***********************
8   REM
9   REM INSTALL TURTLE BEFORE LOADING
10  !
20  ! DEMO1...Draw a "squiral"
30  !
40 DA = 15
50 SQUARE = 240: !  Sub.line #
60  !
70  VIEW# 1: PLT# 1
80  CLR : FULL : HIRES
90  CENTER : TURNTO 30
100  HPRINT (100,0)"* Squiral *"
110  FOR N = 1 TO 63
120  DO SQUARE
130 A = A + 1: LEFT DA
140  NEXT N
150  PRINT  CHR$ (7)
160  HPRINT (90,180)"Press Return"
170  GET X$
180  TEXT
190  END
200  ! ......................
210  !
220  !  Subroutine "SQUARE"
230  !
240  FOR J = 1 TO 4: FWD 10 + A: LEFT
     90: NEXT J
250  RETURN
```

END OF LISTING 5

LISTING 6: DEMO2

```
1   REM  ***********************
2   REM *        DEMO2          *
3   REM  *     BY J.B. WARD     *
4   REM  * COPYRIGHT (C) 1986 *
5   REM  * BY MICROSPARC, INC *
6   REM  * CONCORD, MA  01742 *
7   REM  ***********************
8   REM
9   REM  INSTALL TURTLE BEFORE LOA
       DING
100 TRIANGL = 150
110  GOTO 650
120  !
130  !  SUBROUTINE:TRIANGL
140  !
150  HIDE : FWD L2: SHOW
160  LEFT A1
170  FWD L1: LEFT A2
180  FWD L1: LEFT A2
190  FWD L1
200  RETURN
210  !
220  !  SUB. TO DRAW FACE, ETC.
230  !
240  HPRINT (120,80)"O  O"
250  HPRINT (120,89)"/  "; CHR$ (
     92)
260  CENTER
270  JUMP 123,94
280  TURNTO 45: FWD 3
290  LEFT 20: FWD 3: LEFT 20: FWD 3
300  LEFT 5: FWD 6
310  LEFT 25: FWD 3
320  LEFT 20: FWD 2
330  !  FACE DONE; DO REST.
340  JUMP 0,150
350  TURNTO 0: FWD 270
360  JUMP 40,160: TURNTO  - 110
370  FWD 30: HIDE : FWD 25: SHOW
     : FWD 5
380  JUMP 5,135: TURNTO  - 90
390  FOR I = 1 TO 12
400  FWD 6: RIGHT 12.5
410  NEXT I
420  JUMP 5,135
430  FOR I = 1 TO 3
440  TURNTO  - 80
450  FOR J = 1 TO 4
460  FWD 6: RIGHT 40
470  NEXT J: NEXT I
480  !
490  HPRINT (100,125)"SUMMERTIME!"
500  HPRINT (188,1)"PRESS <SPACE>"
510  HPRINT (205,9)"TO STOP."
520  RETURN
530  !
540  !  SUBROUTINE TO STOP ON
550  !  ANY KEYSTROKE.
560  !
570 K2 =  PEEK (49152)
580  IF K1 = K2 THEN  RETURN
590  PLT# 1: VIEW# 1
600  HOME : TEXT
```

```
610  END
620  !
630  !  MAIN PROGRAM
640  !
650 L1 = 28.8
660 L2 = 100
670 A1 = 150
680 A2 = 120
690  FULL : WHITE
700  PLT# 0: CLR : PLT# 1: CLR
710  HIRES
720 P = 0:V = 1
730 S = 5
740  !
750  FOR A = 0 TO 360 STEP S
760  CENTER : TURNTO A
770 P =  NOT P:V =  NOT V
780  PLT# P: VIEW# V
790  DO TRIANGL
800  NEXT A
810  !
820  GOSUB 240
830  PLT# V
840  GOSUB 240
850  !
860 D = 50
870 K1 =  PEEK (49152)
880  GOSUB 570
890  FOR I = 1 TO D: NEXT I
900 V =  NOT V: VIEW# V
910  GOTO 880
```

END OF LISTING 6

LISTING 7: DEMO3

```
1   REM ***********************
2   REM *        DEMO3        *
3   REM *     BY J.B. WARD    *
4   REM * COPYRIGHT (C) 1986 *
5   REM * BY MICROSPARC, INC *
6   REM * CONCORD, MA  01742 *
7   REM ***********************
10   REM  DEMONSTRATION OF TURTLE
         BASIC
20   REM
30   REM  INSTALL TURTLE BEFORE LO
         ADING
40   !
50   !  (1): Draw the big one.
60   VIEW# 1: PLT# 1
70   WHITE : CLR
80   FULL : HIRES
90 L = 180
100  FOR A = 0 TO 359
110  CENTER : TURNTO A
120  FWD L
130  NEXT A
140  !
150  ! (2): Draw the little one.
160  BLACK
170 S = 3:L = 45
180  FOR A = 0 TO 359 STEP S
190  CENTER : TURNTO A
200  FWD L
210  NEXT A
220  WHITE
230  !
240  !
250  ! (3): Message & wait.
260  !
270  CENTER : FWD 0
280  BLACK
290  HPRINT (90,183);
300  FOR N = 1 TO 14
310  HPRINT  CHR$ (127);
320  NEXT N
330  WHITE
340  HPRINT (98,184)"Press Return"
350  PRINT  CHR$ (7)
360  INPUT X$
370  TEXT
380  END
```

END OF LISTING 7

NIBBLE CALCULATOR

by Newton Saiyuen Lee

SECOND FEATURE

Use your Apple

as a full-function scientific calculator. Features include hyperbolic trig functions, six memory registers, and five user-defined programs.

The Nibble Calculator is a super calculator for the Apple // series. It's powerful, flexible and simple to use. Nibble Calculator can hold up to five user programs in memory at the same time. Your program can use the value in the calculator display and pass a value back. Like a hand-held programmable calculator, it carries out a full range of mathematical functions activated by one or two keypresses (see **Figure 1**).

USING NIBBLE CALCULATOR

Built-in Functions and Keys

The built-in function keys are defined in **Table 1**. The function keys calculate trigonometric functions, exponents, logarithms, roots, powers, absolute values, reciprocals, hyperbolic functions and factorials. Other built-in functions and keys are defined below.

<ESC> — Quit Key. When you press <ESC>, you are first asked for confirmation. Then the user-defined programs and the values stored in memory slots 1-6 are saved to disk. If the program is terminated in a different way, the memories and the user-defined programs are not saved.

V — Second Function Selection Key. When this key is pressed, the arrow next to the V, which normally points to the left, points to the right and flashes, indicating that the second functions (those enclosed in square brackets in **Figure 1**) can be selected. For example, 0.5 V S means ARCSIN(0.5). The V key is a toggle key; that is, if V is pressed by mistake, the first function can be restored by pressing V again. For example, 45 V V S means SIN(45).

@ — Degree/Radian Selection Key. The at sign (@) is used to change the mode of operation for trigonometric and inverse trigonometric functions. In DEG mode (the default), entries and answers are all in decimal degrees. In RAD mode, they are all in radians (1 radian = $180/\pi$ degrees).

D — Display Clear Key. D is used to clear the display (i.e., set it to zero).

P — User-Defined Program Execution Key. Press P to run the user-defined programs. Select the program by number. See the Creating Your Own Programs section for more details.

 # — User-Defined Program Programming Key. The pound sign (#) enters the section of Nibble Calculator that allows you to create or modify user-defined programs. See the Creating Your Own Programs section for more details.

K S — Memory Store. K followed by S is used to store the displayed number in the specified memory slot. The old value in the

Newton Saiyuen Lee, Management Information Center, Vincennes University, Vincennes, IN 47591. Nibble Calculator is compatible with DOS 3.3 and ProDOS.

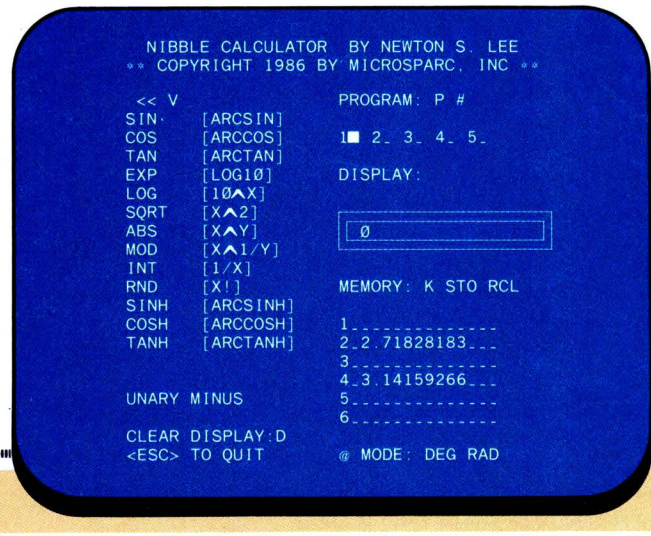


FIGURE 1: Main Calculator Display

slot (if there is one) is erased before the new value is stored in it. For example, 3.5 K S 1 instructs Nibble Calculator to store 3.5 in memory slot 1.

To clear memory slot *n*, simply type **K S** *n* when the displayed number is zero. The display can be set to zero by pressing D. For example, D K S 5 clears memory slot 5.

If the K S key combination is used during a calculation, the result of the calculation is stored in the specified memory slot. For example:

55 + 10 K S 1

puts the result of the calculation (65) into memory slot 1. This is equivalent to typing:

55 + 10 <RETURN> K S 1

When you press the K key, the K just below the display begins to flash. When you press S, the K stops flashing and the S begins flashing. When you press the slot number (1-6) the S stops flashing and the store operation is completed. If you press a key that is not expected, the bell sounds to indicate the error, the flashing stops and the calculator returns to normal entry mode.

K R — Memory Recall. This key sequence is used to retrieve the value stored in the specified memory slot. The value in the slot is not erased. For example, K R 2 retrieves the value from memory slot 2. If the K R key combination is used during a calculation, the number stored in the specified memory slot will be recalled and evaluated with the number in the display. For example: 12 + K R 1 adds the number stored in memory slot 1 to the number 12 (in the display window).

In a manner similar to that described above for the memory store operation, the K and R will flash until the next key in the sequence is pressed.

Numeric Keys. Numeral keys (0-9) are used to enter numbers. The decimal key (.) is used when entering a decimal number. The exponent key (E) is used to enter exponents. For example, 12.5 E 5 signifies 12.5×10^5. The unary minus key (U) is used when entering negative numbers or negative exponents. For example:

U 12.5 = −12.5
1.0 E U 5 = 1.0 E −5

Arithmetic Calculation Keys. Use a plus sign (+) for addition, a minus sign (−) for subtraction, an asterisk (∗) for multiplication, and a slash (/) for division. The priority level is the same for all operators and functions, and they are performed from left to right. For example:

5 + 6 ∗ 5 = 11 ∗ 5 = 55

To get the result of a calculation, you can use either <RETURN> or equals (=).

Since Nibble Calculator doesn't support parentheses, it may be necessary to work from inside an expression out, or to use the memory feature to store an intermediate result. For instance, to get the expected result for the expression above, use the sequence:

6 ∗ 5 + 5 = 35

Entering Your Own Programs in Nibble Calculator

From the main menu (**Figure 1**) press the pound sign (#) to display the programming menu. This is a list of the five available programs. The solid box marks a user program that has already been defined. Let's create a short program and enter it. Press **1**, and you should see TAG = at the top of the screen. Fill in the name for the new program by typing:

ONE-VARIABLE STATISTICAL CALC

and press the <RETURN> key. (The first 70 characters you type are saved.) The following message will be displayed:

ENTER YOUR PROGRAM #1 [3310-3510]
[BLANK LINE = FINISH; Q = QUIT]

You can now enter your new program, just as you would enter an Applesoft program. The line number range for program 1 is 3310 to 3510, as shown inside the brackets. If you try to type in a line number outside this range, you will get the error message:

LINE NUM OUT OF RANGE [3310-3510]
RETYPE LINE

The value shown in the calculator display is available for use by your user program in the variable N. It is automatically passed to this variable when you select a user program. Similarly, if you wish to return a value from your user program to the main calculator, simply assign it to the variable N before returning from your program.

The program can be written like a BASIC program, with the following restrictions:

1. The line number range is limited as specified.
2. The maximum number of lines per program is 100.
3. No comma (,) or colon (:) is allowed.
4. No STOP, END or NEW statement is allowed.
5. You must be very careful with GOTO, THEN, and GOSUB statements, as their misuse can lead to unexpected results and errors.

If you enter a blank line when typing in a new program, you automatically leave the input mode and return to the main calculator program. However, Nibble Calculator keeps your program in memory. Pressing Q lets you quit the user-defined program section and the program you just entered is erased from memory.

Type in the listing in **Example 1** and when you have finished, press the <RETURN> key twice. The disk drive will spin for a while as it creates an EXEC file for your program and you will return to the main calculator display.

Editing Programs

Press the pound sign again to go back to the programming menu. If you press 1, since program 1 already exists, Nibble Calculator puts you automatically into edit mode, and the following message is displayed:

EDIT> L(IST T(AG C(HANGE E(RASE Q(UIT

The edit functions are as follows:

1. L (list) displays a listing of the program (use <CTRL>S to pause)
2. T (tag) changes the TAG of the program
3. C (change) makes modifications to the program
4. E (erase) deletes the program
5. Q quits updating the program (and causes the current program to be discarded)

The functions are self-explanatory except for the change function. When you press C, you will see:

EXAMPLE 1: Sample User-Defined Program

```
3310  DIM X(50)
3315  INPUT "NUMBER OF SAMPLES (2-50)? ";N1
3320  PRINT "ENTER NUMERIC VALUE"
3325  FOR I = 1 TO N1
3330  PRINT I;
3335  INPUT "? ";X(I)
3340  NEXT
3345  FOR I = 1 TO N1
3350 S1 = S1 + X(I)
3355 S2 = S2 + X(I) * X(I)
3360  NEXT
3365  PRINT "TOTAL OF SAMPLES = ";S1
3370  PRINT "SUM OF SQUARES OF SAMPLE = ";S2
3375 S = S1 / N1
3380  PRINT "MEAN VALUE OF SAMPLES = ";S
3385 ST =  SQR ((S2 - N1 * S * S) / (N1 - 1))
3390  PRINT "STANDARD DEVIATION = ";ST
3400 N = ST
```

TABLE 1: Function Keys

Keys	Abbreviation	Function
S	SIN	Sine
C	COS	Cosine
T	TAN	Tangent
VS	ARCSIN	Arcsine
VC	ARCCOS	Arccosine
VT	ARCTAN	Arctangent
X	EXP	Natural exponential (antilogarithm base e)
VL	LOG10	Antilogarithm base 10
Q	SQRT	Square root
VQ	X^2	Square
A	ABS	Absolute value
VA	X^Y	X to the Y power
M	MOD	A MOD B
VM	X^1/Y	Yth root of X
I	INT	Integer truncation e.g., 4.88 I = 4
VI	1/X	Reciprocal
R	RND	Random number between zero and one
VR	X!	Factorial
N	SINH	Hyperbolic sine
O	COSH	Hyperbolic cosine
H	TANH	Hyperbolic tangent
VN	ARCSINH	Inverse hyperbolic sine
VO	ARCCOSH	Inverse hyperbolic cosine
VH	ARCTANH	Inverse hyperbolic tangent

CHANGE> ENTER LINE(S): [3310-3510]
[BLANK LINE = FINISH; Q = QUIT]

You change a line in very much the same way that you enter a new one. Type the line number of the line you want to change and then the line with the changes you want. If you want to add a new line to your program, type the line number and the new line. Enter a blank line when you are through making changes.

Running Programs

Press the P key to display the run menu, which is nearly identical to the programming menu. The solid block marks programs that have been entered. If you try to run a program that has not yet been entered, the error message PROGRAM *#n* IS EMPTY is displayed. To run a program, simply type the program number. Press **1** to run ONE-VARIABLE STATISTICAL CALC. If there are any errors or typing mistakes, an appropriate message is displayed, and you are returned to the main calculator display. Choose the pound sign option and edit the program using the C option.

Error Messages

When you are using the normal functions on Nibble Calculator, three kinds of errors may occur:

1. DIVISION BY ZERO — This error occurs when you try to divide a number by zero. For example: 5/0 <RETURN>
2. ILLEGAL QUANTITY — This error occurs when you try to: *(a)* Apply LOG to a negative number or zero; *(b)* Apply SQR to a negative number; *(c)* Use X^Y when X is negative and Y is not an integer; or *(d)* Apply ARCTANH (inverse hyperbolic tangent) to a number whose absolute value is greater than 1.
3. OVERFLOW — This error occurs when the result of a calculation is too big. You will receive this error message if you enter, for example, 10 E 50 * 20 E 50.

To clear the error message from the display, simply press <RETURN>.

TABLE 2: Program Functions

Lines	Function
90	Dimensions A$, LA%, UA%, P%, T$ and M
100	Defines the line number range for user-defined programs
110-130	Initializes NIBBLE.CALC
140-160	Set MODE = DEG, PI = 3.141592654, and display value = 0
170-450	Handle first functions requested by user
460-470	SIN function
480-490	COS function
500-510	TAN function
520	INT function
530	RND function
540	ABS function
550	SQR function
560	EXP function
570	LOG function
580-620	MOD function
630-650	SINH function
660-680	COSH function
690-710	TANH function
720-850	Handle second function requested by user
860-890	ARCSIN function
900-930	ARCCOS function
940-950	ARCTAN function
960	1/X
970-980	X!
990-1030	X^Y
1040	X^2
1050	LOG10
1060	10^X
1070-1110	X^(1/Y)
1120-1140	ARCSINH function
1150-1170	ARCCOSH function
1180-1200	ARCTANH function
1210-1230	Clear display function (D)
1240-1280	Memory-in/memory-out functions (K STO/RCL)
1290-1360	Arithmetic functions
1370	Error trap for saving memory
1380-1440	Display DV$ and set new number
1450-1480	Display DV$
1490-1510	Erase left arrow and display right arrow
1520-1570	Erase right arrow and display left arrow; display angular mode selection
1580-1690	Handle K STO/RCL request as the second function
1700-1890	Accept user input number and put it into TV$
1900-2100	Run user-defined programs
2110-2280	Create and modify user-defined programs
2290-2420	Create new program
2430-2990	Modify old program:
2510-2550	List
2570-2750	Change
2770-2900	Erase
2920	Quit
2940-2990	Tag
3000-3180	Set up EXEC file for the user-defined program
3190-3230	Schedule an EXEC
3270-4480	User-defined program buffer areas
4520-4810	Error handling routine
4820-4950	Read CAL.INFO
4960-5060	Read CAL.MEMORY
5070-5170	Save MEMORY values and user-defined programs
5180-5570	Main calculator display
5580-5620	Store and recall the value of N for use by user-defined programs

Errors may also occur when running the user-defined programs. The line number where the error is found is displayed, along with the error message. To clear the error message and return to the main menu, simply press <RETURN>. (For a detailed description of the error messages, read the *Apple II Reference Manual*.)

ENTERING THE PROGRAM

To key in Nibble Calculator, type **Listing 1** and save it with the command:

```
SAVE NIBBLE.CALC
```

Because of the user-defined program feature, the lines between **lines 3300 and 4480** are not consecutively numbered by tens. For help in entering *Nibble* listings, see "A Welcome to New *Nibble* Readers" at the beginning of this issue.

HOW IT WORKS

Table 2 documents each section of the program. Some unusual features are explained below in more detail.

Permanent External Files

Two external files must be present for Nibble Calculator to work properly:

1. CAL.INFO contains the values of P%(*n*) and T$(*n*), where 1<=*n*<=5. If this file is not present when NIBBLE.CALC is run, it is automatically created, using zeros and null strings to fill the two arrays.
2. CAL.MEMORY contains the values of M(N) where 1<=N<=6 (see **Table 3**). If this file is not present when NIBBLE.CALC is run, it is automatically created, using zeros to fill the array.

User-Defined Program EXEC Files

A maximum of six EXEC files can be created by NIBBLE.CALC.

TABLE 3: Variable Usage

Variable	Function
A$(*m*)	Program line for the user-defined program (maximum number of lines per program = *m*)
C$	Command character typed by the user
DV	Decimal equivalent of DV$
DV$	Displayed value in the display window
E%	Error flag (1 if an arithmetic error occurs; 0 otherwise)
E$	Contains DOS error messages
EC%	E counter (1 if you type in E as part of the number; 0 otherwise)
K%	KSTO/KRCL flag (0 if KRCL is requested; 1 if KSTO is requested)
LA%(*n*)	Lowest line number for the user-defined program *n*
M(*l*)	The number stored in memory slot *l*
M%	Angular mode indicator (0 if MODE = DEG; 1 if MODE = RAD)
N	The value that can be passed to user programs and returned from user programs
N$	The memory slot number typed in by the user
P%(*n*)	User-defined program existence indicator (1 if user-defined program *n* exists; 0 otherwise)
PC%	Period (.) counter (1 if the user types in a period as part of a real number; 0 otherwise)
T$(*n*)	The TAG for user-defined program *n*
TV$	Temporary value for arithmetic calculations
UA%(*n*)	Upper line limit for user program *n*

Note: The maximum value of *m* is 100, *n* is 5 and *l* is 6.

An EXEC file is used to put the user-defined program into memory. The file name convention is CAL.*n*, where *n* is the program number. Every EXEC file is constructed as follows:

Statement		Meaning
DEL	LA%, UA%	Delete line numbers LA% to UA%
ln	*statement*	User-defined program lines, where *ln*
.	. . .	is the line number
.	. . .	
RUN		

When such a file is EXECed, the buffer area for the particular user-defined program is cleared and refilled with the new program lines. Finally, the RUN command re-initiates NIBBLE.CALC.

User-Defined Program Buffer Areas

The buffer areas are reserved for storing user-defined programs in memory along with NIBBLE.CALC. Each buffer area has the following structure:

Line	Statement
ln	CLEAR : GOSUB 5600
ln	User-defined program statements
. . .	. . .
ln	. . .
ln	GOTO 4490

The first and last lines are fixed and built into the program.

The line number ranges for the user-defined programs are:

User Program	Lines
1	3310-3510
2	3550-3750
3	3790-3990
4	4030-4230
5	4270-4470

The program editor does not allow lines outside the allocated line range.

Passing a Value to a User-Defined Program

When you run a user-defined program, the CLEAR command is executed, so that NIBBLE.CALC's variables will not interfere with variables in the user-defined program. The routine at **line 5580** assigns N the value that was shown in the calculator display by PEEKing memory locations starting at 768. This area is used by the main program to store the contents of DV$ before calling a user program in the following format: length byte, ASCII values of the characters in DV$, duplicate length byte. The final length byte acts as a check to ensure that NIBBLE.CALC only reads valid data from this area of memory. When your program ends, a GOTO 4490 clears ERRFLG, converts the value of N to a string, POKEs the string into memory at location 768 and reinitiates NIBBLE.CALC.

LISTING 1: NIBBLE.CALC

```
10  REM ************************
20  REM *     NIBBLE.CALC      *
30  REM *  BY NEWTON S. LEE    *
40  REM * COPYRIGHT (C) 1986  *
50  REM * BY MICROSPARC, INC  *
60  REM * CONCORD, MA  01742  *
70  REM ************************
```

```
80 D$ =  CHR$ (4):UL$ =  CHR$ (95):LB$ =  CHR$
     (91)
90  DIM A$(100),LA%(5),UA%(5),P%(5),T$(5),M(6
     )
100 LA%(1) = 3310:UA%(1) = 3510:LA%(2) = 3550
     :UA%(2) = 3750:LA%(3) = 3790:UA%(3) = 39
     90:LA%(4) = 4030:UA%(4) = 4230:LA%(5) =
     4270:UA%(5) = 4470
110  GOSUB 4850: REM - READ INFO -
120  GOSUB 4990: REM - READ MEMORY -
130  GOSUB 5210: REM - DISPLAY -
140 M% = 0
150 PI = 3.141592654
160  GOSUB 5590: GOTO 1410
170  REM
180  REM  - BEGIN -
190  REM
200  GOSUB 1520: GOSUB 1730: REM -GET TV$-
210  IF  LEN (TV$) > 0 THEN DV$ = TV$
220 K% = 0: ONERR  GOTO 4580
230  IF C$ = "V" THEN  GOSUB 1490: GET C$: GOTO
     720
240  IF C$ = "S" THEN 460
250  IF C$ = "C" THEN 480
260  IF C$ = "T" THEN 500
270  IF C$ = "I" THEN 520
280  IF C$ = "R" THEN 530
290  IF C$ = "A" THEN 540
300  IF C$ = "Q" THEN 550
310  IF C$ = "X" THEN 560
320  IF C$ = "L" THEN 570
330  IF C$ = "M" THEN 580
340  IF C$ = "N" THEN 630
350  IF C$ = "O" THEN 660
360  IF C$ = "H" THEN 690
370  IF C$ = "D" THEN 1210
380  IF C$ = "K" THEN 1240
390  IF C$ = "P" THEN  GOSUB 5110: GOSUB 1930
     : GOSUB 5210: GOTO 1410
400  IF C$ = "#" THEN  GOSUB 5110: GOSUB 2140
     : GOSUB 5210: GOTO 1410
410  IF C$ = "+" OR C$ = "-" OR C$ = "*" OR C
     $ = "/" THEN 1290
420  IF C$ = "=" OR C$ =  CHR$ (13) THEN 200
430  IF C$ =  CHR$ (27) THEN 5100
440  IF C$ = "@" THEN M% = M% + 1:M% =  INT (
     (M% / 2 -  INT (M% / 2)) * 2 + .05)
450  PRINT  CHR$ (7): GOTO 200
460  IF M% = 0 THEN DV$ =  STR$ ( VAL (DV$) *
     PI / 180)
470 DV$ =  STR$ ( SIN ( VAL (DV$))): GOTO 141
     0
480  IF M% = 0 THEN DV$ =  STR$ ( VAL (DV$) *
     PI / 180)
490 DV$ =  STR$ ( COS ( VAL (DV$))): GOTO 141
     0
500  IF M% = 0 THEN DV$ =  STR$ ( VAL (DV$) *
     PI / 180)
510 DV$ =  STR$ ( TAN ( VAL (DV$))): GOTO 141
     0
520 DV$ =  STR$ ( INT ( VAL (DV$))): GOTO 141
     0
530 DV$ =  STR$ ( RND ( VAL (DV$))): GOTO 141
     0
540 DV$ =  STR$ ( ABS ( VAL (DV$))): GOTO 141
     0
550 DV$ =  STR$ ( SQR ( VAL (DV$))): GOTO 141
     0
560 DV$ =  STR$ ( EXP ( VAL (DV$))): GOTO 141
     0
570 DV$ =  STR$ ( LOG ( VAL (DV$))): GOTO 141
     0
580  GOSUB 1730: IF  LEN (TV$) = 0 THEN TV$ =
     DV$
590  IF C$ = "K" THEN  GOSUB 1610
600 DV =  VAL (DV$):TV =  VAL (TV$):DV$ =  STR$
     ( INT ((DV / TV -  INT (DV / TV)) * TV +
     .05) *  SGN (DV / TV))
610  IF K% = 1 THEN  GOSUB 1680: GOTO 1410
620  GOTO 1450
630 DV =  VAL (DV$)
640 DV$ =  STR$ (( EXP (DV) -  EXP ( - DV)) /
     2)
650  GOTO 1410
660 DV =  VAL (DV$)
```

continued on next page

LISTING 1: NIBBLE.CALC *(continued)*

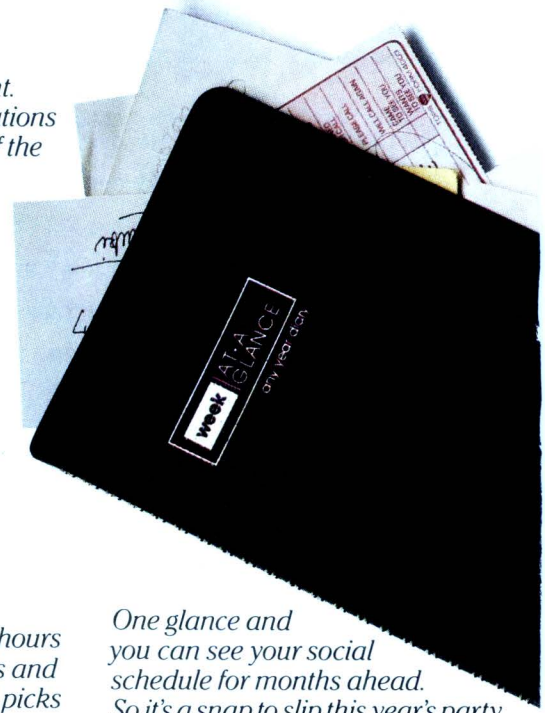
```
670 DV$ =  STR$ (( EXP (DV) +  EXP ( - DV)) /
    2)
680  GOTO 1410
690 DV =  VAL (DV$)
700 DV$ =  STR$ ( -  EXP ( - DV) / ( EXP (DV)
     +  EXP ( - DV)) * 2 + 1)
710  GOTO 1410
720  IF C$ = "S" THEN 860
730  IF C$ = "C" THEN 900
740  IF C$ = "T" THEN 940
750  IF C$ = "I" THEN 960
760  IF C$ = "R" THEN 970
770  IF C$ = "A" THEN 990
780  IF C$ = "Q" THEN 1040
790  IF C$ = "X" THEN 1050
800  IF C$ = "L" THEN 1060
810  IF C$ = "M" THEN 1070
820  IF C$ = "N" THEN 1120
830  IF C$ = "O" THEN 1150
840  IF C$ = "H" THEN 1180
850  GOTO 200
860 DV =  VAL (DV$): IF DV = 1 THEN DV$ =  STR$
    (1.570796327): GOTO 880
870 DV$ =  STR$ ( ATN (DV /  SQR ( - DV * DV +
    1)))
880  IF M% = 0 THEN DV$ =  STR$ ( VAL (DV$) *
    180 / PI)
890  GOTO 1410
900 DV =  VAL (DV$): IF DV = 1 THEN DV$ =  STR$
    (0): GOTO 920
910 DV$ =  STR$ ( -  ATN (DV /  SQR ( - DV *
    DV + 1)) + 1.5708)
920  IF M% = 0 THEN DV$ =  STR$ ( VAL (DV$) *
    180 / PI)
930  GOTO 1410
940 DV$ =  STR$ ( ATN ( VAL (DV$))): IF M% =
    0 THEN DV$ =  STR$ ( VAL (DV$) * 180 / P
    I)
950  GOTO 1410
960 DV$ =  STR$ (1 /  VAL (DV$)): GOTO 1410
970 DV = 1: FOR I = 1 TO  VAL (DV$):DV = DV *
    I: NEXT :DV$ =  STR$ (DV)
980  GOTO 1410
990  GOSUB 1730: IF  LEN (TV$) = 0 THEN TV$ =
    DV$
1000  IF C$ = "K" THEN  GOSUB 1610
1010 DV$ =  STR$ ( VAL (DV$) ^  VAL (TV$))
1020  IF K% = 1 THEN  GOSUB 1680: GOTO 1410
1030  GOTO 1450
1040 DV$ =  STR$ ( VAL (DV$) *  VAL (DV$)): GOTO
    1410
1050 DV$ =  STR$ ( LOG ( VAL (DV$)) /  LOG (1
    0)): GOTO 1410
1060 DV$ =  STR$ (10 ^  VAL (DV$)): GOTO 1410
1070  GOSUB 1730: IF  LEN (TV$) = 0 THEN TV$ =
    DV$
1080  IF C$ = "K" THEN  GOSUB 1610
1090 DV$ =  STR$ ( VAL (DV$) ^ (1 /  VAL (TV$
    )))
1100  IF K% = 1 THEN  GOSUB 1680: GOTO 1410
1110  GOTO 1450
1120 DV =  VAL (DV$)
1130 DV$ =  STR$ ( LOG (DV +  SQR (DV * DV +
    1)))
1140  GOTO 1410
1150 DV =  VAL (DV$)
1160 DV$ =  STR$ ( LOG (DV +  SQR (DV * DV -
    1)))
1170  GOTO 1410
1180 DV =  VAL (DV$)
1190 DV$ =  STR$ ( LOG ((1 + DV) / (1 - DV)) /
    2)
1200  GOTO 1410
1210  HTAB 22: FOR I = 1 TO 15: PRINT " ";: NEXT

1220  HTAB 22: PRINT "0";:DV$ = "0"
1230  GOTO 200
1240 COL% =  POS (0): VTAB 14: HTAB 28: FLASH
    : PRINT "K";: NORMAL : VTAB 11: HTAB COL
    % + 1: GET C$:COL% =  POS (0): VTAB 14: HTAB
    28: INVERSE : PRINT "K";: NORMAL : VTAB
    11: HTAB COL% + 1: IF C$ <  > "S" AND C$
     <  > "R" THEN  PRINT  CHR$ (7);: GOTO 2
    00
```

```
1250 COL% =  POS (Ø): VTAB 14: HTAB 3Ø + 4 *
     (C$ = "R"): FLASH : PRINT C$;: NORMAL : VTAB
     11: HTAB COL% + 1: GET N$: VTAB 14: HTAB
     3Ø + 4 * (C$ = "R"): INVERSE : PRINT C$;
     : NORMAL : VTAB 11: HTAB COL% + 1: IF N$
      < "1" OR N$ > "6" THEN  PRINT  CHR$ (7)
     ;: GOTO 2ØØ
1260  IF C$ = "S" THEN M( VAL (N$)) =  VAL (D
     V$)
1270  IF C$ = "R" THEN DV$ =  STR$ (M( VAL (N
     $)))
1280  GOSUB 552Ø: GOTO 141Ø
1290 TC$ = C$: GOSUB 173Ø: IF  LEN (TV$) = Ø THEN
     TV$ = DV$
1300  IF C$ = "K" THEN  GOSUB 161Ø
1310  IF TC$ = "+" THEN DV$ =  STR$ ( VAL (DV
     $) +  VAL (TV$))
1320  IF TC$ = "-" THEN DV$ =  STR$ ( VAL (DV
     $) -  VAL (TV$))
1330  IF TC$ = "*" THEN DV$ =  STR$ ( VAL (DV
     $) *  VAL (TV$))
1340  IF TC$ = "/" THEN DV$ =  STR$ ( VAL (DV
     $) /  VAL (TV$))
1350  IF K% = 1 THEN  GOSUB 168Ø: GOTO 141Ø
1360  GOTO 145Ø
1370  CALL  - 3288: POKE 216,Ø: HOME : VTAB 1
     2: PRINT "ERROR #" PEEK (222)" IN LINE "
      PEEK (218) + 256 *  PEEK (219): PRINT :
      INPUT "DO YOU WANT TO TRY AGAIN? ";YN$:
      ON (YN$ = "Y") GOTO 511Ø: RETURN
1380  REM
1390  REM  - DISPLAY DV$ -
1400  REM
1410 PC% = Ø:EC% = Ø: VTAB 11: HTAB 22: PRINT
     DV$;
1420  IF  LEN (DV$) = 15 THEN 2ØØ
1430  FOR I = 1 TO 15 -  LEN (DV$): PRINT " "
     ;: NEXT
1440  GOTO 2ØØ
1450 PC% = Ø:EC% = Ø: VTAB 11: HTAB 22: PRINT
     DV$;
1460  IF  LEN (DV$) = 15 THEN 22Ø
1470  FOR I = 1 TO 15 -  LEN (DV$): PRINT " "
     ;: NEXT
1480  HTAB 22: GOTO 22Ø
1490 COL% =  POS (Ø)
1500  VTAB 4: HTAB 2: PRINT "  ";: HTAB 7: FLASH
     : PRINT ">>";: NORMAL : VTAB 11: HTAB CO
     L% + 1
1510  RETURN
1520 COL% =  POS (Ø)
1530  VTAB 4: HTAB 2: INVERSE : PRINT "<<";: NORMAL
     : HTAB 7: PRINT "  ";
1540  IF M% = Ø THEN  VTAB 23: HTAB 28: INVERSE
     : PRINT "DEG";: NORMAL : PRINT " RAD";
1550  IF M% = 1 THEN  VTAB 23: HTAB 28: NORMAL
     : PRINT "DEG ";: INVERSE : PRINT "RAD";
1560  NORMAL : VTAB 11: HTAB COL% + 1
1570  RETURN
1580  REM
1590  REM   - HANDLE K STO/RCL REQ -
1600  REM
1610 COL% =  POS (Ø): VTAB 14: HTAB 28: FLASH
     : PRINT "K";: NORMAL : VTAB 11: HTAB COL
     % + 1: GET C$:COL% =  POS (Ø): VTAB 14: HTAB
     28: INVERSE : PRINT "K";: NORMAL : VTAB
     11: HTAB COL% + 1: IF C$ <  > "R" AND C$
      <  > "S" THEN  PRINT  CHR$ (7);:K% = Ø:
      RETURN
1620 COL% =  POS (Ø): VTAB 14: HTAB 3Ø + 4 *
     (C$ = "R"): FLASH : PRINT C$;: NORMAL : VTAB
     11: HTAB COL% + 1: GET N$: VTAB 14: HTAB
     3Ø + 4 * (C$ = "R"): INVERSE : PRINT C$;
     : NORMAL : VTAB 11: HTAB COL% + 1: IF N$
      < "1" OR N$ > "6" THEN  PRINT  CHR$ (7)
     ;:K% = Ø: RETURN
1630  IF C$ = "S" THEN K% = 1:C$ =  CHR$ (13)
     : RETURN
1640  IF C$ = "R" THEN TV$ =  STR$ (M( VAL (N
     $))):K% = Ø:C$ =  CHR$ (13): RETURN
1650  REM
1660  REM  - HANDLE K STO -
```

continued on page 51

AppleWorks

Only Pinpoint adds pictures to any AppleWorks document. So you can create party invitations that are truly exciting. Even if the weather isn't.



Don't spend hours addressing envelopes and misspelling names. Pinpoint picks up names and addresses from invitations (or any documents) and drops them, letter perfect, onto envelopes. Even if you don't always spell those tongue-twisting names right, at least you'll be consistent.

One glance and you can see your social schedule for months ahead. So it's a snap to slip this year's party in between the garlic festival and your breakdancing lessons.

Jot down important thoughts whenever they strike. The note pad is perfect for reminding you of things you can't do without. Or things you can.

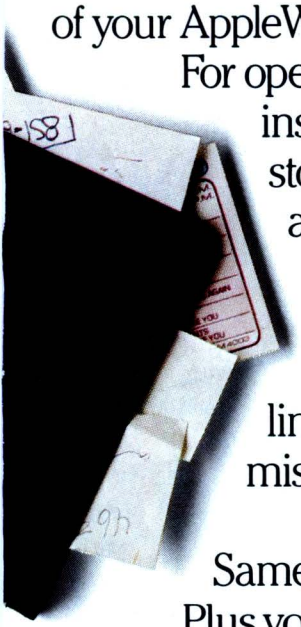
Impress a friend with a personal invitation. It's easy. The automatic dialer finds phone numbers displayed anywhere on the screen, not just in special files.

And it re-dials busy numbers automatically. So you're next in line as soon as your friend's broker hangs up.

Accessories. $69.

Nine accessories to be precise. And in no time, they'll be so much a part of your AppleWorks routine, you'll think of them as built-in, not added-on. Why?

For openers, these handy desktop timesavers "pop-up" via windows from inside AppleWorks *(and most other ProDOS programs).* So you can stop what you're doing to jot down a note, address a letter, or call a friend.



You can even figure out some figures with our calculator. Tap data bases or other computers through our communications window. Even whip out short memos, forms and labels with our line-at-a-time typewriter. And then continue AppleWorks without missing a beat. Or a byte.

What's more, Pinpoint's as easy to use as AppleWorks itself. Same commands. Same hardware.

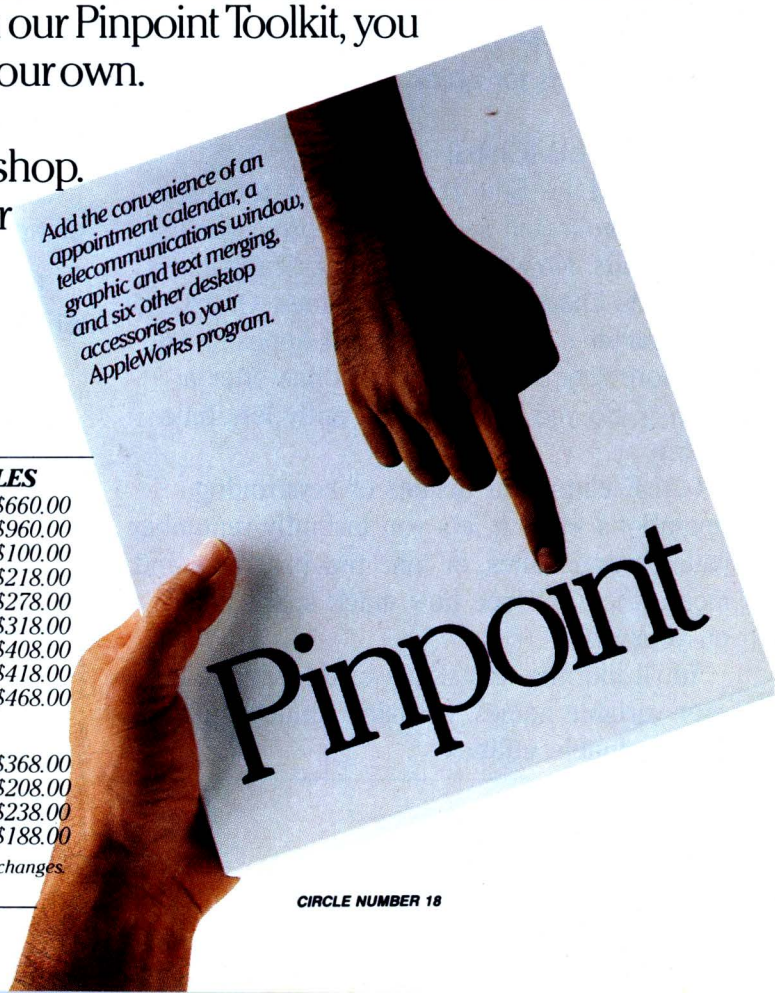
Plus you can boost Pinpoint's performance by adding an Apple Extended Memory Card, the new, high-capacity UniDisk 3.5 floppy, a hard disk, a modem, or one of many popular printers. And with our Pinpoint Toolkit, you can even add desktop accessories of your own.

Want more?

Then visit your local computer shop.

Or call (415) 530-1726 for a dealer near you. You'll see that Pinpoint does more than give AppleWorks a handful of accessories.

It gives AppleWorks the works.



SPECIAL PURCHASE— MAXIMUM PERFORMANCE BUNDLES

10MB SIDER Hard Disk, Siderware, plus Pinpoint for only	*$660.00*
20MB SIDER Hard Disk, Siderware, plus Pinpoint for only	*$960.00*
Pinpoint, plus 256K RAM for RamWorks/Multi-Ram/Apple	*$100.00*
64K RAMWORKS, AppleWorks Expander Software, plus Pinpoint	*$218.00*
64K RAMWORKS, AppleWorks Expander, Pinpoint, plus 256K RAM	*$278.00*
512K RAMWORKS, AppleWorks Expander Software, plus Pinpoint	*$318.00*
1Meg RAMWORKS, AppleWorks Expander Software, plus Pinpoint	*$408.00*
256K Z-RAM, AppleWorks Expander Software, plus Pinpoint	*$418.00*
512 Z-RAM, AppleWorks Expander Software, plus Pinpoint	*$468.00*

FAMOUS BRAND MODEMS, Bundled with Pinpoint

Prometheus 1200A, 1200 Baud External	*IIc/IIe*/MAC*/IBM**	*$368.00*
Prometheus 300c, Compact 300 Baud External	*Mounts on the IIc*	*$208.00*
Volksmodem 1200, 1200 Baud External	*IIc/IIe*/MAC*/IBM**	*$238.00*
ZOOM IIe, 300 Baud Internal	*IIe/Enhanced IIe*	*$188.00*

**Requires $10-15 cable and/or serial interface card, Subject to availability and price changes. Most items available from your dealer.*

CIRCLE NUMBER 18

If This Powerful Applesoft Editor Were Any Simpler, Your Keyboard Could Look Like This.

NEW! NEW!

Announcing a **new** 128K version of ProDOS GALE for the //c and the 128K 80-column //e. GALE.PRO.128 loads itself into auxiliary RAM, totally freeing up your Applesoft program and variable space for your long programs! Current owners of GALE can receive the new version by submitting $10.00 (postpaid) along with the cover page of your original GALE documentation. The upgraded disk will include updated DOS 3.3 versions that will run on all Apple // series computers, plus the new 128K ProDOS version and a 64K ProDOS version for all 80-column Apple //e's.

Meet the word processor *for writing programs,* complete with HELP Screens for quick reference.

With GALE (Global Applesoft Line Editor), single keys stand for long, complicated sequences of programming commands. Now you can automatically insert or delete changes in program lines. Or change a character or word wherever it appears. And no more retyping the rest of a line after a change. So you spend significantly less time typing.

GALE eliminates dozens of keystroking operations while it lets you instantly renumber your program lines, or take two programs and merge them. Or see how much space is left on your disk.

You'll like the immediate cross-index of all your variable names, auto-line numbering and other valuable utilities.

GALE! You'll wonder how you ever got along without it.

To order fill out the attached coupon or call 617-371-1660.

MicroSPARC Inc.

I want GALE! Here's my $49.95

Payable in U.S. funds only.

☐ Mastercard ☐ Visa ☐ Check, M. O.

(Mass. residents add 5% sales tax) **POSTPAID**

Name Tel. #

Address

City State Zip

Signature Date

Charge Card # Exp. Date

MicroSPARC Inc., 45 Winthrop St., Concord, MA 01742

Author: Sandy Mossberg
System Requirements: Applesoft compatibility and DOS 3.3
Apple is a registered trademark of Apple Computer, Inc.

LISTING 1: NIBBLE.CALC *(continued)*

```
1670  REM
1680 M( VAL (N$)) =  VAL (DV$): GOSUB 5520
1690  RETURN
1700  REM
1710  REM  - GET TV$ -
1720  REM
1730  VTAB 11: HTAB 22:TV$ = "":PC% = 0:EC% =
      0
1740  GET C$
1750  IF (C$ < "0" OR C$ > "9") AND C$ <  > "
      ." AND C$ <  >  CHR$ (8) AND C$ <  > "E"
      AND C$ <  > "U" THEN  RETURN
1760  IF  LEN (TV$) = 15 THEN 1810
1770  IF (C$ >  = "0" AND C$ <  = "9") OR (C$
      = "." AND PC% = 0 AND EC% = 0) OR (C$ =
      "E" AND EC% = 0) THEN  PRINT C$;:TV$ = T
      V$ + C$: IF  POS (0) = 22 THEN  GOSUB 18
      80:TV$ = C$: PRINT C$;
1780  IF C$ = "U" AND ( LEN (TV$) = 0 OR  RIGHT$
      (TV$,1) = "E") THEN  PRINT "-";:TV$ = TV
      $ + "-": IF  POS (0) = 22 THEN  GOSUB 18
      80:TV$ = "-": PRINT "-";
1790  IF C$ = "E" AND EC% = 0 THEN EC% = 1
1800  IF C$ = "." AND EC% = 0 AND PC% = 0 THEN
      PC% = 1
1810  IF C$ <  >  CHR$ (8) OR  LEN (TV$) = 0 THEN
      1740
1820  PRINT  CHR$ (8);: PRINT " ";: PRINT  CHR$
      (8);
1830  IF  RIGHT$ (TV$,1) = "E" THEN EC% = 0
1840  IF  RIGHT$ (TV$,1) = "." THEN PC% = 0
1850  IF  LEN (TV$) = 1 THEN TV$ = ""
1860  IF  LEN (TV$) > 1 THEN TV$ =  LEFT$ (TV
      $, LEN (TV$) - 1)
1870  GOTO 1740
1880  FOR I = 1 TO 15: PRINT " ";: NEXT : HTAB
      22
1890  RETURN
1900  REM
1910  REM  - RUN PROGRAM -
1920  REM
1930  HOME : VTAB 2: HTAB 15: INVERSE : PRINT
      " RUN PROGRAM ": NORMAL
1940 J = 1
1950  FOR I = 1 TO 5
1960 J = J + 3: VTAB J: HTAB 1
1970  IF P%(I) = 1 THEN  INVERSE : PRINT " ";
      : NORMAL
1980  IF P%(I) = 0 THEN  PRINT " ";
1990  PRINT " PROGRAM ";I
2000  PRINT "  TAG = ";T$(I)
2010  NEXT
2020  PRINT : INVERSE : PRINT " ";: NORMAL : PRINT
      " <ESC> TO QUIT"
2030  VTAB 22: HTAB 1: PRINT "  PROGRAM";LB$;
      "1-5]: ";: GET N$: PRINT N$;
2040  IF N$ =  CHR$ (27) THEN  RETURN
2050 N =  VAL (N$): IF N < 1 OR N > 5 THEN  PRINT
       CHR$ (7);: GOTO 2030
2060  IF P%(N) = 0 THEN 2090
2070  HOME : ONERR  GOTO 4550
2080  ON N GOTO 3300,3540,3780,4020,4260
2090  VTAB 22: HTAB 2: FLASH : PRINT  CHR$ (7
      );" PROGRAM ";N;" IS EMPTY ": NORMAL
2100  FOR I = 1 TO 500: NEXT I: RETURN
2110  REM
2120  REM  - PROGRAMMING ROUTINE -
2130  REM
2140  HOME : INVERSE : VTAB 1: HTAB 15: PRINT
      " PROGRAMMING ": NORMAL
2150 J = 1
2160  FOR I = 1 TO 5
2170 J = J + 3: VTAB J: HTAB 1
2180  IF P%(I) = 1 THEN  INVERSE : PRINT " ";
      : NORMAL
2190  IF P%(I) = 0 THEN  PRINT " ";
2200  PRINT " PROGRAM ";I
2210  PRINT "  TAG = ";T$(I)
2220  NEXT
2230  PRINT : INVERSE : PRINT " ";: NORMAL : PRINT
      " <ESC> TO QUIT"
2240  VTAB 22: HTAB 1: PRINT "  PROGRAM";LB$;
      "1-5]: ";: GET N$: PRINT N$;
2250  IF N$ =  CHR$ (27) THEN  RETURN
2260 N =  VAL (N$): IF N < 1 OR N > 5 THEN  PRINT
       CHR$ (7);: GOTO 2240
2270  IF P%(N) = 0 THEN  GOSUB 2320: GOTO 303
      0
2280  IF P%(N) = 1 THEN  GOSUB 2460: RETURN
2290  REM
2300  REM  - ENTER NEW PROGRAM -
2310  REM
2320  HOME : INPUT "TAG = ";T$(N): IF  LEN (T
      $(N)) > 70 THEN T$(N) =  LEFT$ (T$(N),70
      )
2330  HOME : INVERSE : PRINT "ENTER YOUR PROG
      RAM #";N;": ";LB$;"";LA%(N);"-";UA%(N);"
      ]"
2340  PRINT LB$;" BLANK LINE = FINISH; Q = QU
      IT ]": NORMAL
2350 I = 1:P%(N) = 1
2360  INPUT "";A$(I)
2370  IF  LEFT$ (A$(I),1) = "Q" THEN  RUN
2380  IF  LEN (A$(I)) = 0 THEN  RETURN
2390  IF  VAL (A$(I)) < LA%(N) OR  VAL (A$(I)
      ) > UA%(N) THEN  PRINT  CHR$ (7);"LINE N
      UM OUT OF RANGE ";LB$;"";LA%(N);"-";UA%(
      N);"]": PRINT "RETYPE LINE:":I = I - 1
2400  IF I = 100 THEN  RETURN
2410 I = I + 1
2420  GOTO 2360
2430  REM
2440  REM  - EDIT OLD PROGRAM -
2450  REM
2460  HOME
2470  VTAB 1: HTAB 1: INVERSE : PRINT "EDIT>
      L(IST T(AG C(HANGE E(RASE Q(UIT ";: NORMAL
      : GET C$: PRINT C$;
2480  IF C$ <  > "L" THEN 2560
2490  HOME
2500  ON N GOTO 2510,2520,2530,2540,2550
2510  LIST 3310,3510: GOTO 2470
2520  LIST 3550,3750: GOTO 2470
2530  LIST 3790,3990: GOTO 2470
2540  LIST 4030,4230: GOTO 2470
2550  LIST 4270,4470: GOTO 2470
2560  IF C$ <  > "C" THEN 2760
2570  HOME : INVERSE : PRINT "CHANGE> ENTER L
      INE(S): ";LB$;"";LA%(N);"-";UA%(N);"]"
2580  PRINT LB$;" BLANK LINE = FINISH; Q = QU
      IT ]": NORMAL
2590  GOSUB 2350
2600  PRINT D$;"OPEN CAL." + N$
2610  PRINT D$;"CLOSE CAL." + N$: PRINT D$;"D
      ELETE CAL." + N$
2620  PRINT D$;"OPEN CAL." + N$
2630  PRINT D$;"WRITE CAL." + N$
2640  PRINT "DEL " +  STR$ (LA%(N)) + "," +  STR$
      (UA%(N))
2650  IF N = 1 THEN  LIST 3310,3510
2660  IF N = 2 THEN  LIST 3550,3750
2670  IF N = 3 THEN  LIST 3790,3990
2680  IF N = 4 THEN  LIST 4030,4230
2690  IF N = 5 THEN  LIST 4270,4470
2700  FOR J = 1 TO I
2710  PRINT A$(J)
2720  NEXT
2730  PRINT "RUN"
2740  PRINT D$;"CLOSE CAL." + N$
2750  GOTO 3220
2760  IF C$ <  > "E" THEN 2910
2770  HOME : PRINT "ERASE> READY TO ERASE PRO
      GRAM #";N: PRINT "PRESS <RETURN> TO CONF
      IRM DELETION": PRINT "   OR <ESC> TO REJ
      ECT DELETION"
2780  GET C$
2790  IF C$ <  >  CHR$ (13) AND C$ <  >  CHR$
      (27) THEN  PRINT  CHR$ (7);: GOTO 2770
2800  PRINT C$;
2810  IF C$ =  CHR$ (27) THEN  HOME : GOTO 24
      70
2820 D$ =  CHR$ (4)
2830  PRINT D$;"OPEN CAL." + N$
2840  PRINT D$;"CLOSE CAL." + N$: PRINT D$;"D
      ELETE CAL." + N$
2850 P%( VAL (N$)) = 0:T$( VAL (N$)) = ""
2860  PRINT D$;"OPEN CAL.INFO"
2870  PRINT D$;"WRITE CAL.INFO"
2880  FOR I = 1 TO 5: PRINT P%(I): PRINT T$(I
      ): NEXT
```

continued on next page

LISTING 1: NIBBLE.CALC *(continued)*

```
2890  PRINT D$;"CLOSE CAL.INFO"
2900  RUN
2910  IF C$ <  > "Q" THEN 2930
2920  RETURN
2930  IF C$ <  > "T" THEN  PRINT  CHR$ (7);: GOTO
     2470
2940  HOME : INPUT "TAG = ";T$(N)
2950  PRINT D$;"OPEN CAL.INFO"
2960  PRINT D$;"WRITE CAL.INFO"
2970  FOR L = 1 TO 5: PRINT P%(L): PRINT T$(L
     ): NEXT
2980  PRINT D$;"CLOSE CAL.INFO"
2990  GOTO 2470
3000  REM
3010  REM  - SAVE PROGRAM -
3020  REM
3030  PRINT D$;"OPEN CAL." + N$
3040  PRINT D$;"CLOSE CAL." + N$: PRINT D$;"D
     ELETE CAL." + N$
3050  PRINT D$;"OPEN CAL." + N$
3060  PRINT D$;"WRITE CAL." + N$
3070  PRINT "DEL " +  STR$ (LA%(N)) + "," +  STR$
     (UA%(N))
3080  FOR J = 1 TO I - 1
3090  PRINT A$(J)
3100  NEXT
3110  PRINT "RUN"
3120  PRINT D$;"CLOSE CAL." + N$
3130  PRINT D$;"OPEN CAL.INFO"
3140  PRINT D$;"WRITE CAL.INFO"
3150  FOR J = 1 TO 5
3160  PRINT P%(J): PRINT T$(J)
3170  NEXT
3180  PRINT D$;"CLOSE CAL.INFO"
3190  REM
3200  REM  -- SCHEDULE EXEC --
3210  REM
3220  PRINT D$;"EXEC CAL." + N$
3230  END
3270  REM
3280  REM  - BUFFER AREA BEGINS -
3290  REM  -1-
3300  CLEAR : GOSUB 5590
3520  GOTO 4490
3530  REM  -2-
3540  CLEAR : GOSUB 5590
3760  GOTO 4490
3770  REM  -3-
3780  CLEAR : GOSUB 5590
4000  GOTO 4490
4010  REM  -4-
4020  CLEAR : GOSUB 5590
4240  GOTO 4490
4250  REM  -5-
4260  CLEAR : GOSUB 5590
4480  GOTO 4490
4490  PRINT : PRINT "PRESS ANY KEY TO CONTINU
     E ";: GET K$
4500  PRINT K$;
4510  POKE 216,0: GOSUB 5620: RUN
4520  REM
4530  REM  - HANDLE ERROR -
4540  REM
4550 L =  PEEK (218) +  PEEK (219) * 256
4560  PRINT  CHR$ (7);"ERROR IN LINE NUMBER "
     ;L
4570  GOSUB 4600: PRINT E$: GOTO 4790
4580  HTAB 21: GOSUB 4600: PRINT  CHR$ (7); LEFT$
     (E$,16);: GET C$
4590  HTAB 21: FOR I = 1 TO 16: PRINT " ";: NEXT
     : HTAB 22: CALL  - 3288: GOTO 160
4600 E =  PEEK (222)
4610  IF E = 0 THEN E$ = "NEXT WITHOUT FOR"
4620  IF E = 16 THEN E$ = "SYNTAX ERROR      "
4630  IF E = 22 THEN E$ = "RETURN WITHOUT GOS
     UB"
4640  IF E = 42 THEN E$ = "OUT OF DATA       "
     : REM 6 SPACES
4650  IF E = 53 THEN E$ = "ILLEGAL QUANTITY"
4660  IF E = 69 THEN E$ = "OVERFLOW         ":
     REM 8 SPACES
4670  IF E = 77 THEN E$ = "OUT OF MEMORY    "
4680  IF E = 90 THEN E$ = "UNDEFINED STATEMEN
     T"
4690  IF E = 107 THEN E$ = "BAD SUBSCRIPT   "
4700  IF E = 120 THEN E$ = "REDIMENSIONED ARR
     AY"
4710  IF E = 133 THEN E$ = "DIVISION BY ZERO"
4720  IF E = 163 THEN E$ = "TYPE MISMATCH   "
4730  IF E = 176 THEN E$ = "STRING TOO LONG"
4740  IF E = 191 THEN E$ = "FORMULA TOO COMPL
     EX"
4750  IF E = 224 THEN E$ = "UNDEFINED FUNCTIO
     N"
4760  IF E = 254 THEN E$ = "BAD RESPONSE TO A
     N INPUT STATEMENT"
4770  POKE 216,0
4780  RETURN
4790  PRINT : PRINT : PRINT "PRESS ANY KEY TO
      CONTINUE ";: GET K$
4800  PRINT K$
4810  RUN
4820  REM
4830  REM  - READ INFO -
4840  REM
4850  ONERR  GOTO 4940
4860  PRINT D$;"OPEN CAL.INFO"
4870  PRINT D$;"READ CAL.INFO"
4880  FOR J = 1 TO 5
4890  INPUT P%(J)
4900  INPUT T$(J)
4910  NEXT
4920  PRINT D$;"CLOSE CAL.INFO"
4930  RETURN
4940  CALL  - 3288: POKE 216,0: PRINT D$"CLOS
     E CAL.INFO": PRINT D$"DELETE CAL.INFO": IF
      PEEK (222) = 5 THEN  PRINT D$"OPEN CAL.
     INFO": PRINT D$"WRITE CAL.INFO": FOR I =
     1 TO 5: PRINT 0: PRINT "": NEXT : PRINT
     D$"CLOSE CAL.INFO": GOTO 4850
4950  HOME : VTAB 12: PRINT "ERROR #" PEEK (2
     22)" IN LINE " PEEK (218) +  PEEK (219) *
     256: END
4960  REM
4970  REM  - READ MEMORY -
4980  REM
4990  ONERR  GOTO 5050
5000  PRINT D$;"OPEN CAL.MEMORY"
5010  PRINT D$;"READ CAL.MEMORY"
5020  FOR I = 1 TO 6: INPUT M(I): NEXT
5030  PRINT D$;"CLOSE CAL.MEMORY"
5040  RETURN
5050  CALL  - 3288: POKE 216,0: PRINT D$"CLOS
     E CAL.MEMORY": PRINT D$"DELETE CAL.MEMOR
     Y": IF  PEEK (222) = 5 THEN  PRINT D$"OP
     EN CAL.MEMORY": PRINT D$"WRITE CAL.MEMOR
     Y": FOR I = 1 TO 6: PRINT 0: NEXT : PRINT
     D$"CLOSE CAL.MEMORY": GOTO 4990
5060  HOME : VTAB 12: PRINT "ERROR #" PEEK (2
     22)" IN LINE " PEEK (218) + 256 *  PEEK
     (219): END
5070  REM
5080  REM -SAVE MEMORY & PROGS-
5090  REM
5100  HOME : VTAB 10: INPUT "ARE YOU SURE YOU
      WANT TO QUIT? ";YN$: ON YN$ <  > "Y" GOTO
     130: GOSUB 5110: GOTO 5170
5110  ONERR  GOTO 1370
5120  PRINT : PRINT D$;"OPEN CAL.MEMORY"
5130  PRINT D$;"WRITE CAL.MEMORY"
5140  FOR I = 1 TO 6: PRINT M(I): NEXT
5150  PRINT D$;"CLOSE CAL.MEMORY"
5160  PRINT D$;"SAVE NIBBLE.CALC": POKE 216,0
     : RETURN
5170  HOME : END
5180  REM
5190  REM  - CALC DISPLAY -
5200  REM
5210  HOME
5220  PRINT " ";: INVERSE : PRINT " NIBBLE CA
     LCULATOR ";: NORMAL : PRINT " BY NEWTON
     S. LEE": PRINT "** COPYRIGHT 1986 BY MIC
     ROSPARC, INC **"
5230  PRINT : INVERSE : HTAB 5: PRINT "V";: NORMAL

5240  PRINT : INVERSE : PRINT "S";: NORMAL : PRINT
     "IN   ";LB$;"ARCSIN]"
5250  INVERSE : PRINT "C";: NORMAL : PRINT "O
     S   ";LB$;"ARCCOS]"
```

continued on page 54

1792k 16-Bit IIe!

Why pay more for a lesser card that works in 8-Bit just because it's advertised a lot? You can buy Checkmate Technology's **State-Of-The-Art MULTIRAM IIe**™ that works great (100%) in 8-Bit, has a true Co-Processor port, & optional 16-Bit 65C816 slot saver Co-Processor card. We've lowered many prices until 1-15-86 & we'll sell you Jeeves™ at $29, Pinpoint™ at $49, or Supercalc 3A™ at $124, **WITH EACH 576k OR LARGER MULTIRAM CARD.**

- **MULTIRAM IIe IS A DIRECT SUBSTITUTE FOR RAMWORKS II**™ or Apple Ext 80 column cards. **MULTIRAM RUNS ALL (100%) 3rd PARTY SOFTWARE** that the others do & its PAL circuit allows for changing memory mapping protocols too!
- **UP TO 768k MAIN BOARD MEMORY, UP TO 1024K PIGGYBACK BOARD MEMORY WITH FREE RGB,** sharp 80 columns, Double Hi-Res Graphics, no slot-1 interference! **A TOTAL OF 1792k MEMORY AT A MUCH LOWER PRICE!**
- **OPTIONAL 16-Bit 65C816 CO-PROCESSOR CARD** that plugs into MULTIRAM using no slots! TRULY STATE-OF-THE ART, BUT IT CAN'T RUN ON RAMWORKS!
- **FREE APPLEWORKS TIME & DATE ON-SCREEN SOFTWARE** that can auto-date and auto-time stamp database files with any ProDOS clock.
- **FREE ULTRA-FAST RAM DISK SOFTWARE** that isn't like Ramworks™ $29 Ram disk software. It can be run alone or WITH APPLEWORKS. FREE RAM TEST & optional CP/M & Pascal Ram disk too!
- **FREE APPLEWORKS EXPANDER SOFTWARE** that modifies AppleWorks once so all features are automatic, loads ALL or PARTS of AppleWorks into memory, runs 20 x faster, increases the Desktop over 1125k, auto-segments large files onto multiple disks, stores over 5325 records! An increased Clipboard/Word Processor update, 16-Bit integrated software AND more piggyback memory is due soon.

- **15 DAY MONEY BACK GUARANTEE, FREE SOFTWARE UPDATES, FREE 64k MEMORY WITH EACH 256k/512k CARD** ONLY FROM COIT VALLEY COMPUTERS. 5 YEAR WARRANTY THAT, UNLIKE RAMWORKS, INSURES COVERAGE NO MATTER WHERE YOU BOUGHT IT! CALL FOR CURRENT PRICES, QUANTITY DISCOUNTS **OR NEW FEATURES!**

	LOWER OUR ~~LOW~~ PRICE
64k **MULTIRAM** IIe	155.
128k **MULTIRAM** IIe	173.
320k MULTIRAM IIe	187.
576k MULTIRAM IIe	237.
768k MULTIRAM IIe	287.
1024k **MULTIRAM** IIe/FREE RGB	388.
1280k **MULTIRAM** IIe/FREE RGB	499.
1536k **MULTIRAM** IIe/FREE RGB	548.
1792k **MULTIRAM** IIe/FREE RGB	598.
64k Memory Expander Chips (8)	20.
256k Memory Expander Chips (8)	60.
Pico™ Slimline Drive IIc, IIe, II+	165.
Apple IIe Enhancement Kit	62.
65C816 Co-Processor Card	162.
CP/M or Pascal Ram Disk (each)	17.
Switch-A-Slot	165.

CALL FOR OTHER APPLE PERIPHERAL PRICES.

640k 16-Bit IIc!

Checkmate Technology's **State-Of-The-Art IIc** cards easily expand your IIc up to 640k, are 100% compatable with all IIc software/hardware, & come with the same **FREE SOFTWARE** as MULTIRAM IIe. **MULTIRAM C** is non-upgradable and **MULTIRAM CX** can be upgraded with a **16-Bit 65C816 Co-Processor kit! CALL FOR CURRENT PRICING & QUANTITY DISCOUNTS!**

- **NO JUMPER WIRES, CLIPS, OR DRIVE REMOVAL REQUIRED FOR INSTALLATION. ALL CHIPS ARE SOCKETED AND REMOVABLE** - unlike the competition.
- **USES ABOUT 50% LESS POWER** than the competition-causing less power supply strain or battery pack drain!
- **15 DAY MONEY BACK SATISFACTION GUARANTEE** from Coit Valley Computers. 5 YEAR WARRANTY THAT, UNLIKE THE COMPETITION, INSURES COVERAGE NO MATTER WHERE YOU BOUGHT IT!
- **LOWER PRICES** - We sell IIc cards for much less, and our software updates are free, while competitors usually charge at least $10.

	LOWER OUR ~~LOW~~ PRICE
256k MULTIRAM C	229.
512k MULTIRAM C	279.
IIc Battery Pack	179.
C-VEU Flat Panel Display	449.
256k MULTIRAM CX	278.
512k MULTIRAM CX	318.
16-Bit 65C816 Kit	135.

($10 less if bought with card)

Terms: Add $4-Ground or $6-Air shipping & phone # to all U.S. orders (foreign orders extra). Add 3% for P.O.'s (3% 7 net 30) & MasterCard/Visa (include #/expir). For fastest delivery send Cashier's/Certified check, Money Order. C.O.D. (add $5) & personal checks accepted (allow 14 days). Tex res add 6⅛% tax. **CALL FOR LATEST PRICES & QUANTITY DISCOUNTS!**

COIT VALLEY COMPUTERS **(214) 234-5047**
14055 Waterfall Way Dallas, Texas 75240

LISTING 1: NIBBLE.CALC *(continued)*

```
5260  INVERSE : PRINT "T";: NORMAL : PRINT "A
      N    ";LB$;"ARCTAN]"
5270  PRINT "E";: INVERSE : PRINT "X";: NORMAL
      : PRINT "P    ";LB$;"LOG10]"
5280  INVERSE : PRINT "L";: NORMAL : PRINT "O
      G    ";LB$;"10^X]"
5290  PRINT "S";: INVERSE : PRINT "Q";: NORMAL
      : PRINT "RT   ";LB$;"X^2]"
5300  INVERSE : PRINT "A";: NORMAL : PRINT "B
      S    ";LB$;"X^Y]"
5310  INVERSE : PRINT "M";: NORMAL : PRINT "O
      D    ";LB$;"X^1/Y]"
5320  INVERSE : PRINT "I";: NORMAL : PRINT "N
      T    ";LB$;"1/X]"
5330  INVERSE : PRINT "R";: NORMAL : PRINT "N
      D    ";LB$;"X!]"
5340  PRINT "SI";: INVERSE : PRINT "N";: NORMAL
      : PRINT "H   ";LB$;"ARCSINH]"
5350  PRINT "C";: INVERSE : PRINT "O";: NORMAL
      : PRINT "SH   ";LB$;"ARCCOSH]"
5360  PRINT "TAN";: INVERSE : PRINT "H";: NORMAL
      : PRINT "   ";LB$;"ARCTANH]"
5370  PRINT : PRINT : INVERSE : PRINT "U";: NORMAL
      : PRINT "NARY MINUS"
5380  PRINT : PRINT "CLEAR DISPLAY:";: INVERSE
      : PRINT "D";: NORMAL
5390  PRINT : PRINT "<ESC> TO QUIT"
5400  VTAB 4: HTAB 20: PRINT "PROGRAM: ";: INVERSE
      : PRINT "P";: NORMAL : PRINT " ";: INVERSE
      : PRINT "#";: NORMAL
5410  VTAB 6: HTAB 20
5420  FOR I = 1 TO 5: PRINT "";I;
5430  IF P%(I) = 1 THEN  INVERSE : PRINT " ";
      : NORMAL
5440  IF P%(I) = 0 THEN  PRINT UL$;
5450  PRINT " ";: NEXT
5460  VTAB 8: HTAB 20: PRINT "DISPLAY:": PRINT

5470  INVERSE : HTAB 20: FOR I = 20 TO 38: PRINT
      " ";: NEXT : PRINT
5480  HTAB 20: PRINT " ";: HTAB 38: PRINT " "
5490  HTAB 20: FOR I = 20 TO 38: PRINT " ";: NEXT
      : PRINT : NORMAL
5500  PRINT : HTAB 20: PRINT "MEMORY:";
5510  HTAB 28: INVERSE : PRINT "K";: HTAB 30:
      PRINT "S";: NORMAL : PRINT "TO ";: INVERSE
      : PRINT "R";: NORMAL : PRINT "CL"
5520  VTAB 16: FOR I = 1 TO 6: HTAB 20: PRINT
      "";I;
5530  FOR II = 1 TO 15: PRINT UL$;: NEXT
5540  IF M(I) <  > 0 THEN  HTAB 22: PRINT M(I
      );
5550  PRINT : NEXT
5560  PRINT : HTAB 20: INVERSE : PRINT "@";: NORMAL
      : PRINT " MODE: DEG RAD";
5570  RETURN
5580  REM  READ DV$ VALUE FORM PAGE 3
5590  DV$ = "":L =  PEEK (768): IF L < 1 OR L >
      13 OR L <  >  PEEK (769 + L) THEN N = 0:
      DV$ = "0": RETURN
5600  FOR I = 1 TO L:DV$ = DV$ +  CHR$ ( PEEK
      (768 + I)): NEXT : POKE 768,255:N =  VAL
      (DV$): RETURN
5610  REM  STORE DV$ ON PAGE 3
5620  DV$ =  STR$ (N):L =  LEN (DV$): POKE 768
      ,L: FOR I = 1 TO L: POKE 768 + I, ASC ( MID$
      (DV$,I,1)): NEXT : POKE 769 + L,L: RETURN
```

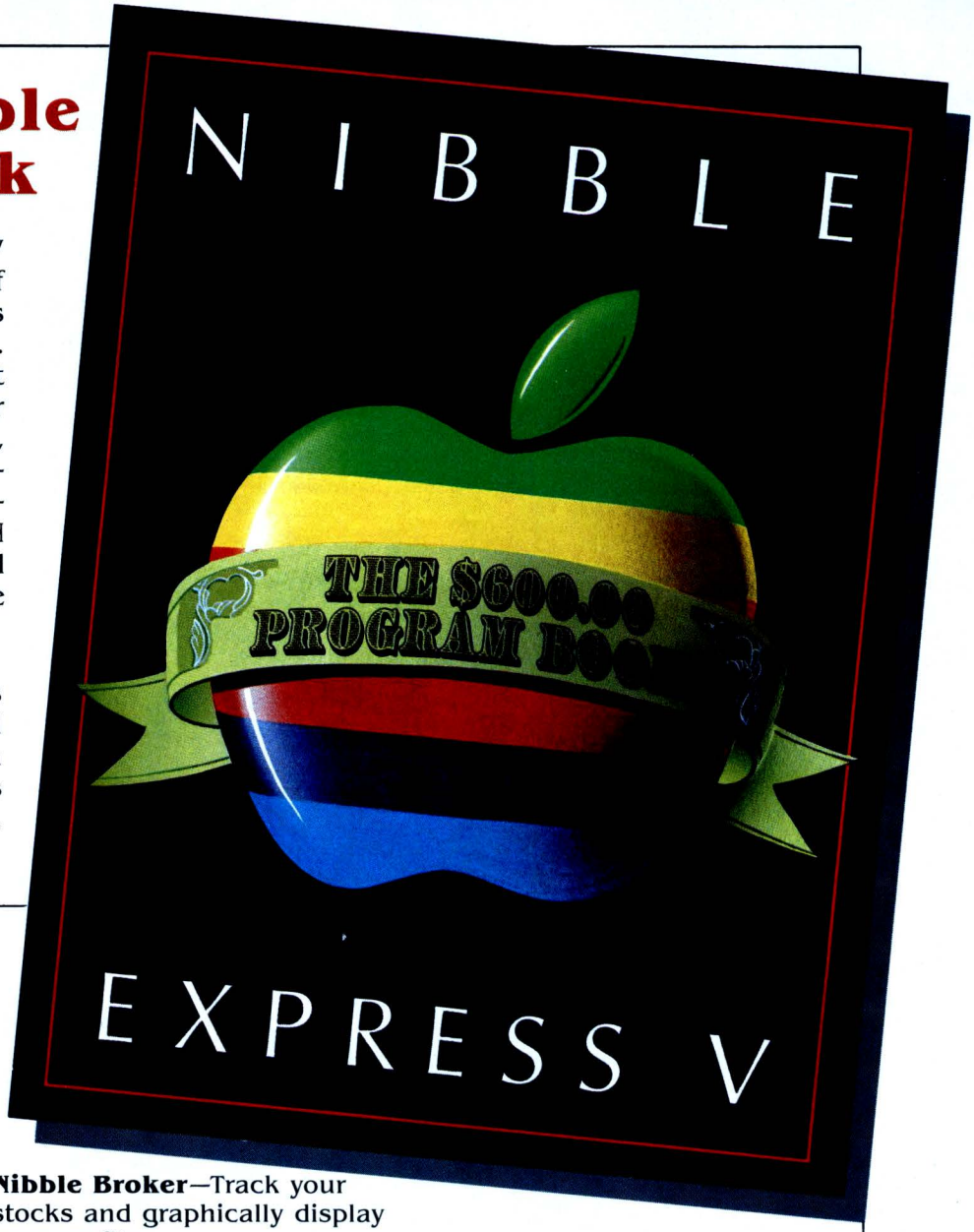
END OF LISTING 1

```
            KEY PERFECT 5.0
              RUN ON
            NIBBLE.CALC
=========================================
  CODE-5.0    LINE# - LINE#    CODE-4.0
  --------    ------------     --------
  8E1DC7C1      10 -   100       A618
  9FB85120     110 -   200       498B
  48BD11BF     210 -   300       4ABE
  30C3962A     310 -   400       5398
  E30A5B6E     410 -   500       87F0
  490DE3EE     510 -   600       835C
  F104043F     610 -   700       493B
  093F2542     710 -   800       3BD6
  CC6C06D0     810 -   900       5DE2
  7AA67F35     910 -  1000       7158
  0248DB98    1010 -  1100       6DB4
  C5C34956    1110 -  1200       4400
  A60B4434    1210 -  1300       C067
  59D00C7A    1310 -  1400       8DF3
  CAB19AB7    1410 -  1500       6269
  972ABA1C    1510 -  1600       5087
  A1F51793    1610 -  1700       B4A6
  679AE5F0    1710 -  1800       A5C3
  00084F67    1810 -  1900       5339
  9AB1FECF    1910 -  2000       3E68
  A39649F5    2010 -  2100       711B
  E277F43E    2110 -  2200       41B1
  E3392B7E    2210 -  2300       5C9B
  3DCF984F    2310 -  2400       9FC9
  8FE42444    2410 -  2500       4D31
  732871F1    2510 -  2600       7284
  3111BC1A    2610 -  2700       64FB
  B9E0EFEE    2710 -  2800       65EB
  DBE0DAFB    2810 -  2900       6548
  2F76E076    2910 -  3000       4B63
  5DAAB77E    3010 -  3100       583F
  990BACF5    3110 -  3200       49D1
  2D92D342    3210 -  3540       2B20
  98D704C9    3760 -  4480       257A
  7AF4F848    4490 -  4580       59BD
  ADD6A696    4590 -  4680       83E2
  A4BD0D2D    4690 -  4780       7CC0
  0F2EFD83    4790 -  4880       3C04
  20CC186E    4890 -  4980       8264
  EF325A6C    4990 -  5080       9BBD
  590A76CC    5090 -  5180       6129
  A834E213    5190 -  5280       881E
  56080564    5290 -  5380       A14D
  3A77E5E9    5390 -  5480       5FDD
  FEA6A2AF    5490 -  5580       682E
  0A7700CB    5590 -  5620       5DD8
  352E16D5 = PROGRAM TOTAL =     2961
```

The $600 Apple Program Book

The Nibble Express Volume V contains over $600* worth of ready-to-type Apple programs from Nibble Magazine 1984. It's an anthology of the best articles and programs for Home, Education, Business, and Entertainment with updated listings and enhancements. It also contains Tips 'N Techniques, Utilities, and other goodies for getting more out of your Apple.

Even if you own all 12 issues of Nibble from 1984, you'll want the Express V so you can have updated program listings in one convenient package. It's a *must* for your library.



Nibble Express V Includes:

Nibble File Cabinet—Flexible Database Management System that lets you create variable length records.

Applesoft Global Editor—Powerful Editor for Applesoft programs offers selected or automatic search and replace.

Sound Synthesizer—Sound Creation Utility that lets you "draw" your sound effects and simple melodies on the Hi-Res screen.

RAM Disk 64—Create a 170 sector pseudodisk on your 128K IIe or IIc for fast access with normal DOS 3.3 commands.

Coupman—Make better use of your coupons and save!

Nibble Diner—Personal Diet Planner provides nutritional analysis of foods and keeps track of your eating habits.

Nibble Garage—Improves car care by tracking maintenance and generating reminders of necessary repairs.

Nibble Broker—Track your stocks and graphically display your profits.

Hi-Res Houdini—Machine Language Graphics Utility creates special graphics effects.

The BASIC Assembler—Editor/Assembler package, written in Applesoft BASIC, produces machine language programs quickly and easily.

*$600 commercial value if purchased on disk.

☐ **Send me Nibble Express V! Here's my $19.95**

Also send me:
☐ Nibble Express Vol. I at $14.95 ea.
☐ Nibble Express Vol. II at $14.95 ea.
☐ Nibble Express Vol. III at $17.95 ea.
☐ Nibble Express Vol. IV at $17.95 ea.
Payable in U.S. funds only.

Please add $1.75 shipping/handling per copy (outside U.S. add: $2.75 Surface or $6.50 Air Mail per copy). Mass. residents add 5% sales tax.

☐ MasterCard
☐ Visa
☐ Check, M.O.

Name

Address

City State Zip

Charge Card # Exp. Date

Signature Tel. #

45 Winthrop St., Concord, MA 01742 (617) 371-1660

TANK COMBAT

FEATURED GAME

by Rudy A. Guy

This Hi-Res *graphics strategy game pits you against the computer in deadly tank combat. You control six tanks to defend your headquarters.*

It happened one cold, winter day — my joystick died. I couldn't shoot invading aliens or defend space ports. My favorite action games were useless, as blank as the empty monitor screen. I stared at it dully. A snowy Sunday afternoon is no time to lose your joystick. In desperation I turned on the television.

The local cable station was running a vintage World War II movie. Real war. Footage of tanks overrunning the French countryside and soldiers marching in formation. The tanks were everywhere — barrelling down country lanes, plowing through thick hedges, lurching over ditches. Tanks blowing up, tanks firing ammunition at an invisible enemy. Then it came to me — Tank Combat, the tank warfare of the 21st century!

Tank Combat is a strategy game that pits you against the computer in an all-out struggle. You control six tanks that are about to be attacked by the invading computer-controlled tanks. Your goal is to wipe out all the computer tanks before they annihilate your forces. A game in progress is shown in **Figure 1**. Tank Combat does not use paddles or

Rudy A. Guy, 1114 West 22nd St., Erie, PA 16502. *Tank Combat is compatible with DOS 3.3 and ProDOS.*

SEARCH AND DESTROY TYPOS IN THE PROGRAMS YOU ENTER FROM MAGAZINES*

In order to eliminate the typos in your programs you've got to find them first. And that can mean hours of wasted time spent in proofreading. Character by character, line by line.

But now you can spotlight these typing errors in the programs you enter from books and magazines. Instantly, on your video screen or printer.

Key Perfect generates a "Check Code Table." Match it against the table published with the original program and you can quickly and easily detect entry errors.

Key Perfect generates a precision check code for:

- Each 10 lines of BASIC (Applesoft and Integer).
- Each 80 Bytes of Machine Language (binary file).

Key Perfect is also the perfect companion program to microSPARC's Global Applesoft Line Editor (GALE) or Nibble's Machine Language Editor (MLE), which feature word processing techniques for correcting your programs.

Key Perfect promises to make your programs perfect—even if your typing isn't!

New! ProDOS Key Perfect! Works with both DOS 3.3 and ProDOS disks automatically. Plus, a new, improved, more accurate checksum algorithm! Generates tables with old and new codes.

Current Key Perfect owners:
Upgrade to the new ProDOS version by sending $10.00 (postpaid) plus the cover page from your original Key Perfect manual.

** Key Perfect tables are currently published in Nibble magazine and several User Group newsletters.*

MicroSPARC Inc.

Send me Key Perfect!
Here's my $29.95.
Payable in U.S. funds only.

POSTPAID
☐ MasterCard ☐ Visa ☐ Check, M.O.
(Mass. residents add 5% sales tax)

Name _____ Tel. # _____

Address _____

City _____ State _____ Zip _____

Signature _____ Date _____

Charge Card # _____ Exp. Date _____

45 Winthrop St., Concord, MA 01742 (617) 371-1660

joysticks. Leave out the sound routine and you can sneak in games at the office.

PLAYING THE GAME

When TANK.COMBAT is run, the computer draws the 16 by 18 playing grid and places 28 anti-tank barriers on it — 14 in the top half and 14 in the bottom. Next, in the top half of the grid (rows 1-8) the computer places its headquarters (HQ), ammo dump (AD) and fuel depot (FD), and hides six anti-tank mines. The computer's six tanks are then positioned in the top three rows of the grid. Then you can place your HQ, fuel depot, ammo dump and anti-tank mines anywhere on the lower half of the grid (rows 9-16). Your six tanks, however, must start the game in rows 14-16.

Tanks have an armor rating of one to six, with six the most vulnerable and one the strongest. The armor rating is the same as the number of the tank. When all the pieces have been placed on the game grid, the action begins.

Placing Shots

Each round begins when the computer player places its shots in secret locations. Then you place your shots and move your tanks. After the computer moves its tanks, the positions of its shots are revealed one by one, and you are informed of any damage they inflict. Finally, your shots are displayed in the same manner. Tanks and barriers are always destroyed when hit; mines are destroyed randomly at a 50/50 rate; HQ's, ammo dumps, and fuel depots can only be destroyed when overrun by a tank.

Placing a shot is accomplished by first moving the solid square using the I, J, K, and M keys, and then entering its position by pressing the space bar. Shots may be placed anywhere on the board, except where you have already placed shots for that round.

Each player starts with six shots per round, but this number is decreased if the player's HQ or ammo dump is overrun. If the ammo dump is overrun, then the number of shots is reduced to one shot per tank. If the HQ is overrun, the total is cut in half. For instance, a player with four tanks and no ammo dump gets four shots, and a player with an ammo dump and any number of tanks, but no HQ, gets three shots.

Moving Tanks

Each tank that hasn't been destroyed may be moved, starting with your number one tank and ending with number six. The total number of remaining moves is displayed at the bottom of the screen. One tank move is a move in either the horizontal or vertical direction. A diagonal move takes two moves — one horizontal and one vertical. Each tank may move up to three squares in a turn, or you can choose not to move it. Use the I, J, K, and M keys to position the tank and the space bar to complete the move. If the move is illegal because you have exceeded the three-square maximum or the total for that round, you will be given another chance to move that tank. It is also illegal to move a tank to a square occupied by your own tank or an anti-tank barrier. A tank may pass through or jump over tanks or barriers, but may not land on them.

Each player starts with 12 moves per round — two for each tank. When a tank is destroyed, the total is decreased by two moves. When a player's fuel depot is destroyed, then the total becomes one move per tank. When the HQ is destroyed, the number of moves otherwise allowed is cut in half.

Be careful of the enemy's hidden mines on the top half of the board. Also, be sure you don't move a tank onto a square where you have placed a shot.

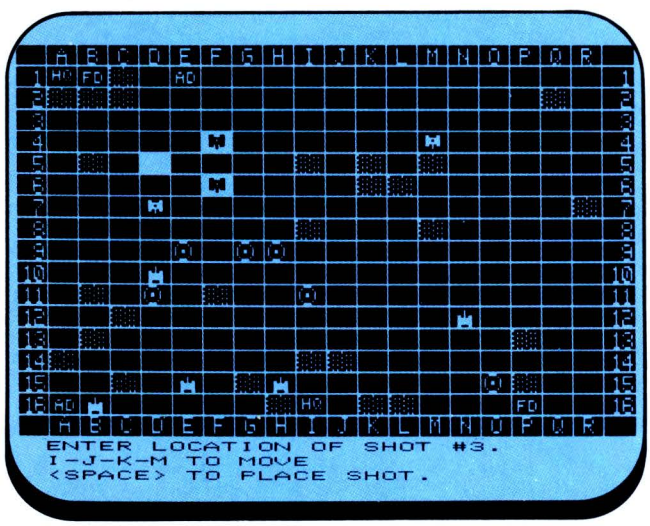
Tank-to-Tank Combat

Another way to battle the opposition forces is to have a player's

TABLE 1: Program Logic

Lines	Function
100	Loads the shape table and initializes the pointers
110-130	Initialization phase of game
170-240	Check the game grid to determine the number of shots and moves for the human player
250-290	Record the player's shot coordinates
300-500	Record the player's tank moves, check for mines, HQ, fuel depot or ammo dump and take appropriate action
510-520	Check for a winner
530-620	Check whether the computer's shots hit the player's tanks
630-670	Check tank positions and determine the outcome of tank-to-tank combat
680	Checks for a winner
690-780	Initiate the computer's move and check the number of shots and spaces allowed
790-850	Enter the computer's shot coordinates for use after the next player's tank move
860-930	Move the tanks on the computer's game grid
940	Checks for a winner
950-1040	Check whether the player's shots hit a computer tank and check for a winner
1060-1100	Check for tank-to-tank combat
1110	Checks for a winner
1120	Starts the next round
1130-1530	Tank-to-tank combat routine
1540-1650	Check for a winner or a tie game, and display the appropriate message
1660-1950	Select computer shots based on player's tank locations
1960-2770	Select computer tank moves based on location of human tanks, HQ, ammo dump and fuel depot
2780-3060	Initialization routine, which draws the game grid and places 14 barriers at random on both halves of the grid
3070-3310	Locate computer's game pieces on the grid
3320-3520	Locate player's game pieces on the grid
3530-3550	Pick random numbers
3560-3570	Draw and erase player's mines at beginning and end of player's turn
3580-3630	Display where shots land on game grid
3640	WAIT loop
3650	Centering routine for messages
3660-3690	POKE sound routine for messages
3700-4100	Input routine
4110-4120	Print I, J, K, M prompt

FIGURE 1: Tank Combat in Progress



tank land on a space adjacent to the enemy's tank (either horizontally or vertically). The survivor of the battle is the tank with the lower armor rating number. If the tanks have the same rating, both are destroyed. Since the armor ratings of the computer's tanks are never displayed, tank-to-tank combat is risky!

An average game lasts 10 to 15 minutes. But, to paraphrase an ad slogan, I'll bet you can't play just one game!

ENTERING THE PROGRAM

To key in the program, type in the Applesoft program shown in **Listing 1** and save it before you run it with the command:

SAVE TANK.COMBAT

To enter TANK.SHAPES (**Listing 2**) first enter the Monitor by typing CALL −151 <RETURN>. Start entering the shape table at $803. When all the code has been entered, save the shape table by typing:

BSAVE TANK.S,A$803,L$278

For more help with entering *Nibble* listings, see "A Welcome to New *Nibble* Readers" at the beginning of this issue.

HOW IT WORKS

TANK.COMBAT (**Listing 1**) relocates itself above Hi-Res page 1. A binary shape table file, TANK.SHAPES (**Listing 2**), resides where Applesoft programs are usually placed.

All of the major routines in TANK.COMBAT have REM statements, but **Table 1** describes the program logic in greater detail.

Turtle BASIC, Nibble Calculator and Tank Combat are available on diskette for an introductory price of $19.95 plus $1.50 shipping/handling ($2.50 outside the U.S.) from Nibble, 45 Winthrop St., Concord, MA 01742. Introductory price expires 3/31/86.

LISTING 1: TANK.COMBAT

```
10  REM ************************
20  REM *      TANK.COMBAT      *
30  REM *      BY RUDY GUY      *
40  REM * COPYRIGHT (C) 1986 *
50  REM * BY MICROSPARC, INC *
60  REM * CONCORD, MA  01742 *
70  REM ************************
80  HOME : VTAB 12: HTAB 8: INVERSE : PRINT "
    TANK COMBAT";: NORMAL : PRINT " BY RUDY
    GUY": PRINT "** COPYRIGHT 1986 BY MICROS
    PARC, INC **"
90  IF  PEEK (104) <  > 64 THEN  POKE 104,64:
     POKE 103,1: POKE 16384,0: PRINT  CHR$ (
    4)"RUN TANK.COMBAT"
100  PRINT  CHR$ (4)"BLOAD TANK.SHAPES,A$803"
    : POKE 232,3: POKE 233,8: ROT= 0: SCALE=
    1
110  GOSUB 3660: REM  INITIALIZE SOUND ROUTIN
    E
120  DIM A$(16,18): TEXT : HOME : VTAB 22:M$ =
    "TANK COMBAT BY RUDY GUY": GOSUB 3650: VTAB
    23:M$ = "* COPYRIGHT 1986 BY MICROSPARC,
     INC *": GOSUB 3650
130  GOSUB 2780:TU = 1: GOSUB 700
140  REM  BEGIN GAME
150 MV = 0:FL = 0:HQ = 0:HS = 0:NU = 10:B$ =
    "C"
160  HOME :M$ = "DISPLAYING HUMAN'S MINES.": GOSUB
    3650
170  FOR I = 1 TO 16: FOR J = 1 TO 18
180  IF A$(I,J) = "MI2" THEN  XDRAW 19 AT J *
    14,Y + (I * 9)
190  IF A$(I,J) = "AD2" THEN HS = 6
200  IF A$(I,J) = "FD2" THEN FL = 1
210  IF A$(I,J) = "HQ2" THEN HQ = 1
```

```
220  IF  RIGHT$ (A$(I,J),1) = "H" THEN MV = M
     V + 1: IF HS <  > 6 THEN HS = HS + 1
230  NEXT J,I: IF FL = 1 THEN MV = MV * 2:FL =
     0
240  IF  NOT HQ THEN HS =  INT (HS / 2): IF H
     S = 0 THEN HS = 1
250  FOR I = 1 TO HS
260 OB$ = "SHOT #" +  STR$ (I): GOSUB 4120:RS
      = 16:RE = 1:R = 1:C = 1:SH = 37:A$ = "K
     ": GOSUB 3710
270 HX(I) = R:HY(I) = C
280  NEXT
290  FOR I = 1 TO HS: XDRAW 37 AT HY(I) * 14,
     Y + (HX(I) * 9): NEXT
300  REM  MOVE TANKS
310  FOR K = 1 TO 6:TX = 0:TY = 0
320  IF XH(K) > 0 THEN TX = XH(K):TY = YH(K)
330  IF TX = 0 THEN 480
340  HOME : INVERSE : VTAB 24: PRINT "MOVES R
     EMAINING:"MV;: NORMAL : VTAB 21: HTAB 1:
      PRINT "NEW LOCATION FOR TANK #"K: PRINT
     "USE I-J-K-M TO MOVE": PRINT "<SPACE> TO
      PLACE TANK";
350  FOR L = 1 TO 11: XDRAW 31 AT TY * 14,Y +
     (TX * 9): FOR D = 1 TO 50: NEXT D,L:SH =
     20 + K: XDRAW SH AT TY * 14,Y + (TX * 9)
360 R = TX:C = TY:RS = 16:RE = 1:MT = 1:A$ =
     "K": GOSUB 3710
370  IF R = TX AND C = TY THEN  XDRAW SH AT T
     Y * 14,Y + (TX * 9): XDRAW 31 AT TY * 14
     ,Y + (TX * 9): GOTO 480
380 MX =  ABS (TX - R):MY =  ABS (TY - C):M =
     MX + MY
390  IF M > 3 OR M > MV THEN  PRINT : PRINT  CHR$
     (7)"YOU CAN'T MOVE THAT FAR!": XDRAW SH AT
     YH(K) * 14,Y + (XH(K) * 9): XDRAW 31 AT
     YH(K) * 14,Y + (XH(K) * 9): GOSUB 3640: GOTO
     340
400  IF A$(R,C) = "HQ" THEN  XDRAW 32 AT C *
     14,Y + (R * 9):A$(R,C) = ""
410  IF A$(R,C) = "AD" THEN  XDRAW 33 AT C *
     14,Y + (R * 9):A$(R,C) = "":
420  IF A$(R,C) = "FD" THEN  XDRAW 34 AT C *
     14,Y + (R * 9):A$(R,C) = ""
430  IF A$(R,C) = "" THEN A$(R,C) =  STR$ (K)
      + "H":XH(K) = R:YH(K) = C:A$(TX,TY) = "
     ": XDRAW SH AT TY * 14,Y + (TX * 9): XDRAW
     31 AT C * 14,Y + (R * 9):TX(K) = R:TY(K)
      = C: GOTO 460
440  IF  LEFT$ (A$(R,C),1) = "M" THEN  XDRAW
     SH AT TY * 14,Y + (TX * 9):A$(R,C) = "":
      FOR D = 1 TO 50: XDRAW 35 AT C * 14,Y +
     (R * 9): NEXT : HCOLOR= 3: DRAW 37 AT C *
     14,Y + R * 9: HCOLOR= 0:XH(K) = 0:YH(K) =
     0:A$(TX,TY) = "": HOME : PRINT  CHR$ (7)
     "TANK HIT MINE AND WAS DESTROYED!": GOSUB
     3640: GOTO 460
450  IF A$(R,C) <  > "" THEN  HOME : PRINT  CHR$
     (7)"THAT SPACE IS OCCUPIED.": XDRAW SH AT
     YH(K) * 14,Y + XH(K) * 9: XDRAW 31 AT YH
     (K) * 14,Y + XH(K) * 9: GOSUB 3640: GOTO
     340
460 MV = MV - M: IF MV = 0 THEN K = 6
470 TX = 0:TY = 0
480  NEXT : HOME
490  FOR I = 9 TO 16: FOR J = 1 TO 18: IF A$(
     I,J) = "MI2" THEN  XDRAW 19 AT J * 14,Y +
     (I * 9)
500  NEXT J,I
510  GOSUB 1540: REM  CHECK FOR A WINNER
520  IF SC = 0 THEN 640
530  FOR I = 1 TO SC
540  IF A$(CX(I),CY(I)) = "" THEN  PRINT "COM
     PUTER SHOT NUMBER "I" MISSED.": GOSUB 35
     80: GOTO 620
550  IF A$(CX(I),CY(I)) = "B" THEN  HCOLOR= 3
     : DRAW 20 AT CY(I) * 14,Y + (CX(I) * 9):
     A$(CX(I),CY(I)) = "": PRINT "COMPUTER SH
     OT NUMBER "I" DESTROYED A       BARRIER."
     : GOSUB 3580: GOTO 620
560  IF  LEFT$ (A$(CX(I),CY(I)),1) = "M" THEN
      GOSUB 3530: IF N / 2 =  INT (N / 2) THEN
      PRINT "COMPUTER SHOT NUMBER "I" MISSED.
     ": GOSUB 3580: GOTO 620
```

continued on next page

LISTING 1: TANK.COMBAT *(continued)*

```
570  IF  LEFT$ (A$(CX(I),CY(I)),1) = "M" THEN
     A$(CX(I),CY(I)) = "": PRINT "COMPUTER SH
     OT NUMBER "I" DESTROYED A MINE.";: GOSUB
     3580: GOTO 620
580  IF  RIGHT$ (A$(CX(I),CY(I)),1) = "H" THEN
      XDRAW 31 AT CY(I) * 14,Y + (CX(I) * 9):
      PRINT "COMPUTER SHOT NUMBER "I" DESTROY
     ED A TANK." CHR$ (7);:NU =  VAL (A$(CX(I
     ),CY(I))):TX(NU) = Ø:TY(NU) = Ø
590  IF  RIGHT$ (A$(CX(I),CY(I)),1) = "H" THEN
      GOSUB 3580:XH( VAL (A$(CX(I),CY(I)))) =
     Ø:YH( VAL (A$(CX(I),CY(I)))) = Ø:A$(CX(I
     ),CY(I)) = "": GOTO 620
600  IF  RIGHT$ (A$(CX(I),CY(I)),1) = "C" THEN
      XDRAW 36 AT CY(I) * 14,Y + (CX(I) * 9):
     XC( VAL (A$(CX(I),CY(I)))) = Ø:YC( VAL (
     A$(CX(I),CY(I)))) = Ø: PRINT "THE COMPUT
     ER SHOT ITS OWN TANK!" CHR$ (7):A$(CX(I)
     ,CY(I)) = "": GOSUB 3580: GOTO 620
610  PRINT "COMPUTER SHOT NUMBER "I" MISSED."
     : GOSUB 3580
620  FOR D = 1 TO 3000: NEXT : GOSUB 1540: NEXT

630  HOME :A = 31:B = 36
640  FOR K = 1 TO 6
650  IF TX(K) = Ø THEN 670
660  GOSUB 1130: REM  CHECK FOR TANK TO TANK
     COMBAT
670 TX(K) = Ø:TY(K) = Ø: NEXT K
680  GOSUB 1540: REM  CHECK FOR A WINNER
690  REM  COMPUTER MOVE
700  HOME :M$ = "COMPUTER PLACING ITS SHOTS."
     : GOSUB 3650
710 MV = Ø:FL = Ø:HQ = Ø:SC = Ø:B$ = "H"
720  FOR I = 1 TO 16: FOR J = 1 TO 18
730  IF A$(I,J) = "AD" THEN SC = 6
740  IF A$(I,J) = "FD" THEN FL = 1
750  IF A$(I,J) = "HQ" THEN HQ = 1
760  IF  RIGHT$ (A$(I,J),1) = "C" THEN MV = M
     V + 1: IF SC <  > 6 THEN SC = SC + 1
770  NEXT J,I: IF FL = 1 THEN MV = MV * 2:FL =
     Ø
780  IF  NOT HQ THEN SC =  INT (SC / 2): IF S
     C = Ø THEN SC = 1
790  REM  PICK COMPUTER SHOTS
800 S = Ø
810  FOR K = 6 TO 1 STEP  - 1
820  IF XH(K) > Ø THEN I = XH(K):J = YH(K):S =
     S + 1: GOSUB 1660: IF S = SC THEN K = 1
830  NEXT
840  IF S < SC THEN 810
850  IF TU THEN TU = Ø: RETURN
860  HOME :M$ = "COMPUTER'S TURN TO MOVE TANK
     S.": GOSUB 3650
870 NU = 4: GOSUB 3530: IF N / 2 =  INT (N /
     2) THEN W = 1: FOR K = 6 TO 1 STEP  - 1:
     TX = Ø:TY = Ø: GOTO 890
880  FOR K = 1 TO 6:TX = Ø:TY = Ø
890  IF XC(K) > Ø THEN I = XC(K):J = YC(K): GOSUB
     1960:MV = MV - S
900  IF W AND MV <  = Ø THEN K = 1
910  IF  NOT W AND MV <  = Ø THEN K = 6
920  REM
930  NEXT K:W = Ø
940  GOSUB 1540: REM  CHECK FOR WINNER
950  HOME : FOR I = 1 TO HS
960  IF A$(HX(I),HY(I)) = "" THEN  PRINT "HUM
     AN'S SHOT NUMBER "I" MISSED.": GOSUB 361
     Ø: GOTO 1040
970  IF A$(HX(I),HY(I)) = "B" THEN  HCOLOR= 3
     : DRAW 20 AT HY(I) * 14,Y + (HX(I) * 9):
     A$(HX(I),HY(I)) = "": PRINT "HUMAN'S SHO
     T NUMBER "I" DESTROYED A": PRINT "BARRIE
     R.": GOSUB 3610: GOTO 1040
980  IF  LEFT$ (A$(HX(I),HY(I)),1) = "M" THEN
      GOSUB 3530: IF N / 2 =  INT (N / 2) THEN
      PRINT "HUMAN'S SHOT NUMBER "I" MISSED."
     : GOSUB 3610: GOTO 1040
990  IF  LEFT$ (A$(HX(I),HY(I)),1) = "M" THEN
     A$(HX(I),HY(I)) = "": PRINT "HUMAN'S SHO
     T NUMBER "I" DESTROYED A MINE.": GOSUB 3
     610: GOTO 1040
```

```
1000  IF  RIGHT$ (A$(HX(I),HY(I)),1) = "C" THEN
      XDRAW 36 AT HY(I) * 14,Y + (HX(I) * 9):
      PRINT "HUMAN'S SHOT NUMBER "I" DESTROYE
     D A TANK." CHR$ (7):NU =  VAL (A$(HX(I),
     HY(I))):TX(NU) = 0:TY(NU) = 0
1010  IF  RIGHT$ (A$(HX(I),HY(I)),1) = "C" THEN
      GOSUB 3610:XC(NU) = 0:YC(NU) = 0:A$(HX(
     I),HY(I)) = "": GOTO 1040
1020  IF  RIGHT$ (A$(HX(I),HY(I)),1) = "H" THEN
      XDRAW 31 AT HY(I) * 14,Y + (HX(I) * 9):
     XH( VAL (A$(HX(I),HY(I)))) = 0:YH( VAL (
     A$(HX(I),HY(I)))) = 0: PRINT "THE HUMAN
     SHOT ITS OWN TANK!" CHR$ (7):A$(HX(I),HY
     (I)) = "": GOSUB 3610: GOTO 1040
1030  PRINT "HUMAN'S SHOT NUMBER "I" MISSED."
     : GOSUB 3610
1040  GOSUB 3640:HX(I) = 0:HY(I) = 0: GOSUB 1
     540: NEXT
1050  HOME
1060  FOR K = 1 TO 6
1070  IF TX(K) = 0 THEN 1100
1080 A = 36:B = 31
1090  GOSUB 1130: REM  CHECK FOR TANK TO TANK
      COMBAT
1100 TX(K) = 0:TY(K) = 0: NEXT K
1110  GOSUB 1540: REM  CHECK FOR A WINNER
1120  HOME : GOTO 150
1130  REM   ROUTINE TO CHECK FOR TANK TO TANK
      COMBAT
1140 D = 1
1150  ON D GOSUB 1180,1240,1300,1360
1160 D = D + 1: IF D = 5 THEN D = 1: RETURN
1170  GOTO 1150
1180 R = TX(K) - 1:C = TY(K): IF R =  < 1 THEN
      RETURN
1190  IF  RIGHT$ (A$(R,C),1) = B$ THEN 1210
1200  RETURN
1210  GOSUB 1420
1220  IF  NOT FL THEN  RETURN
1230 FL = 0:D = 4: RETURN
1240 R = TX(K) + 1:C = TY(K): IF R > 16 THEN
      RETURN
1250  IF  RIGHT$ (A$(R,C),1) = B$ THEN 1270
1260  RETURN
1270  GOSUB 1420
1280  IF  NOT FL THEN  RETURN
1290 FL = 0:D = 4: RETURN
1300 R = TX(K):C = TY(K) - 1: IF C < 1 THEN  RETURN

1310  IF  RIGHT$ (A$(R,C),1) = B$ THEN 1330
1320  RETURN
1330  GOSUB 1420
1340  IF  NOT FL THEN  RETURN
1350 FL = 0:D = 4: RETURN
1360 R = TX(K):C = TY(K) + 1: IF C > 18 THEN
      RETURN
1370  IF  RIGHT$ (A$(R,C),1) = B$ THEN 1390
1380  RETURN
1390  GOSUB 1420
1400  IF  NOT FL THEN  RETURN
1410 FL = 0:D = 4: RETURN
1420  REM  COMBAT
1430  FOR I = 1 TO 5: PRINT  CHR$ (7): NEXT :
      HOME : HTAB 17: FLASH : PRINT "COMBAT":
      NORMAL : FOR I = 1 TO 3000: NEXT
1440  FOR I = 1 TO 10: XDRAW A AT TY(K) * 14,
     Y + (TX(K) * 9): XDRAW B AT C * 14,Y + (
     R * 9): FOR J = 1 TO 50: NEXT : NEXT
1450  IF K =  VAL ( LEFT$ (A$(R,C),1)) THEN  XDRAW
     A AT TY(K) * 14,Y + (TX(K) * 9): XDRAW B
      AT C * 14,Y + (R * 9):FL = 1: HOME : PRINT
     "BOTH TANKS DESTROYED.":A$(R,C) = "":A$(
     TX(K),TY(K)) = "":XC(K) = 0:YC(K) = 0:XH
     (K) = 0:YH(K) = 0: GOTO 1530
1460  IF B$ = "C" THEN 1490
1470  IF K <  VAL ( LEFT$ (A$(R,C),1)) THEN  XDRAW
     36 AT TY(K) * 14,Y + (TX(K) * 9):A$(TX(K
     ),TY(K)) = "":XC(K) = 0:YC(K) = 0: HOME
     : PRINT "COMPUTER TANK DESTROYED!": GOTO
     1530
1480  GOTO 1520
1490  IF K >  VAL ( LEFT$ (A$(R,C),1)) THEN  XDRAW
     36 AT C * 14,Y + (R * 9):XC( VAL (A$(R,C
     ))) = 0:YC( VAL (A$(R,C))) = 0:A$(R,C) =
     "": HOME : PRINT "COMPUTER TANK DESTROYE
     D!": GOTO 1530
```

continued on page 65

Raise your Apple's IQ Twelve Times A Year!

Subscribe Now & Save $12.00 off the Cover Price!

A one-year subscription to NIBBLE brings you twelve issues packed with programs and comprehensive articles to help you get the best out of your Apple.

You'll get over $500 worth of programs for Home, Business, Education and Entertainment with complete instructions. Nibble articles show what each program does, how to use it and enter it into your Apple, Franklin ACE or other Applesoft-compatible computer.

You'll enjoy regular features for the beginner as well as the expert. Among these are the Educational Corner, where programs help make learning fun, Tips & Techniques which showcases little-known programming tricks, Utilities to facilitate Basic, DOS & Printing, and Games with arcade fun you can type and run.

Try a NIBBLE!

Here's what some of our Readers say:

- ☐ *"Certainly the best magazine on the Apple!"*
- ☐ *"Programs remarkably easy to enter."*
- ☐ *"Your service is fantastic . . . as a matter of fact, I'm amazed!"*

Apple® is a registered trademark of Apple Computer, Inc.
ACE® is a registered trademark of Franklin Computer, Inc.

Note
☐ Canada surface subscription rate is $34.95
☐ Outside the U.S. and Canada surface subscription rate is $39.95
☐ Domestic U.S. First Class subscription rate is $51.95
☐ Canada Air Mail subscription rate is $59.95
☐ Outside the U.S. and Canada Air Mail subscription rate is $89.95
All payments must be in U.S. funds drawn on a U.S. bank.

Join the more than 120,000 Apple/ACE users who say:

"NIBBLE is terrific!"

nibble

We accept Master Charge & Visa

45 Winthrop, St., Concord, MA 01742

I'll try nibble!

Enclosed is my: ☐ **$26.95 (for 12 issues — U.S. Only)**
☐ **$49.95 (for 24 issues — U.S. Only)**
☐ **$69.95 (for 36 issues — U.S. Only)**

(Outside U.S. see special note on this page)

☐ **check** ☐ **money order** ☐ **bill me (U.S. only)**

Your subscription will begin with the next issue published after receipt of your check/money order.
I use the following computers:
☐Apple // Family ☐Macintosh

Card # _____ Expires _____
PLEASE PRINT CLEARLY

Signature _____

Name _____

Address _____

City _____

State _____ Zip _____

LISTING 1: TANK.COMBAT *(continued)*

```
1500  XDRAW 31 AT TY(K) * 14,Y + (TX(K) * 9):
      FL = 1: HOME : PRINT "HUMAN'S TANK DESTR
      OYED!":XH( VAL (A$(TX(K),TY(K)))) = Ø:YH
      ( VAL (A$(TX(K),TY(K)))) = Ø:A$(TX(K),TY
      (K)) = ""
1510  GOTO 1530
1520  XDRAW 31 AT C * 14,Y + (R * 9):FL = 1: HOME
      : PRINT "HUMAN'S TANK DESTROYED.":XH( VAL
      (A$(R,C))) = Ø:YH( VAL (A$(R,C))) = Ø:A$
      (R,C) = ""
1530  FOR I = 1 TO 4000: NEXT : RETURN
1540  REM  LOOK FOR A WINNER
1550 H = Ø:C = Ø
1560  FOR WI = 1 TO 6
1570  IF XC(WI) > Ø THEN C = C + 1
1580  IF XH(WI) > Ø THEN H = H + 1
1590  NEXT
1600  IF H = Ø AND C = Ø THEN  FOR I = 1 TO 5
      : PRINT  CHR$ (7): NEXT : PRINT "THIS MA
      TCH HAS ENDED IN A TIE. WOULD YOULIKE TO
       TRY AGAIN ?";: GOTO 1640
1610  IF H = Ø THEN  FOR I = 1 TO 5: PRINT  CHR$
      (7): NEXT : HOME : PRINT "SORRY HUMAN, I
       WON THIS ROUND. WOULD YOULIKE TO TRY AG
      AIN?";: GOTO 1640
1620  IF C = Ø THEN  FOR I = 1 TO 5: PRINT  CHR$
      (7): NEXT : HOME : PRINT "CONGRATULATION
      S, YOU WON THIS ROUND.    WOULD YOU LIKE
       TO TRY AGAIN?";: GOTO 1640
1630  RETURN : REM  STILL TANKS LEFT
1640  INPUT "";A$: IF  LEFT$ (A$,1) = "N" THEN
       TEXT : HOME : END
1650  CLEAR : GOTO 120
1660  REM  SELECT COMPUTER SHOT LOCATION
1670 NU = 10: GOSUB 3530
1680  IF N / 2 =  INT (N / 2) THEN D = 4: GOTO
      1710
1690  IF N = 1 OR N = 3 OR N = 9 THEN NU = 2:
       GOSUB 3530:D = 1 + N: GOTO 1710
1700  IF N = 5 OR N = 7 OR N = 10 THEN D = 1
1710 NU = 3: GOSUB 3530: IF N = 3 THEN NU = 2
      : GOSUB 3530: ON D GOSUB 1860,1910,1910,
      1860: IF DI THEN DI = Ø:NU = 2: GOTO 173
      Ø
1720 NU = 3
1730  GOSUB 3530: ON D GOTO 1740,1770,1800,18
      30
1740  IF I + N > 16 THEN 1670
1750  IF  RIGHT$ (A$(I + N,J),1) <  > "1" AND
       RIGHT$ (A$(I + N,J),1) <  > "C" THEN CX
      (S) = I + N:CY(S) = J: RETURN
1760  GOTO 1670
1770  IF J - N < 1 THEN 1670
1780  IF  RIGHT$ (A$(I,J - N),1) <  > "1" AND
       RIGHT$ (A$(I,J - N),1) <  > "C" THEN CX
      (S) = I:CY(S) = J - N: RETURN
1790  GOTO 1670
1800  IF J + N > 18 THEN 1670
1810  IF  RIGHT$ (A$(I,J + N),1) <  > "1" AND
       RIGHT$ (A$(I,J + N),1) <  > "C" THEN CX
      (S) = I:CY(S) = J + N: RETURN
1820  GOTO 1670
1830  IF I - N < 1 THEN 1670
1840  IF  RIGHT$ (A$(I - N,J),1) <  > "1" AND
       RIGHT$ (A$(I - N,J),1) <  > "C" THEN CX
      (S) = I - N:CY(S) = J: RETURN
1850  GOTO 1670
1860  IF N = 1 THEN 1890
1870  IF J - 1 < 1 THEN  RETURN
1880 J = J - 1:DI = 1: RETURN
1890  IF J + 1 > 18 THEN  RETURN
1900 J = J + 1:DI = 1: RETURN
1910  IF N = 1 THEN 1940
1920  IF I - 1 < 1 THEN  RETURN
1930 I = I - 1:DI = 1: RETURN
1940  IF I + 1 > 16 THEN  RETURN
1950 I = I + 1:DI = 1: RETURN
1960  REM  MOVE COMPUTER TANK
1970 TX = I:TY = J:FL = Ø:TI = Ø:S = Ø
1980  FOR D = 1 TO 16
1990  IF  RIGHT$ (A$(D,J),1) = "H" OR A$(D,J)
      = "HQ2" OR A$(D,J) = "AD2" OR A$(D,J) =
      "FD2" THEN FL = D
2000  NEXT D: IF FL <  > Ø THEN  GOTO 2400
2010  FOR D = 1 TO 18
2020  IF  RIGHT$ (A$(I,D),1) = "H" OR A$(I,D)
      = "HQ2" OR A$(I,D) = "AD2" OR A$(I,D) =
      "FD2" THEN FL = D
2030  NEXT D: IF FL <  > Ø THEN  GOTO 2420
2040 NU = 4: GOSUB 3530:TI = TI + 1: IF TI >
      10 THEN  RETURN
2050 FL = Ø: ON N GOTO 2120,2190,2260,2330
2060  IF A$(TR,TC) = "HQ2" THEN  XDRAW 32 AT
      TC * 14,Y + (TR * 9):A$(TR,TC) = ""
2070  IF A$(TR,TC) = "AD2" THEN  XDRAW 33 AT
      TC * 14,Y + (TR * 9):A$(TR,TC) = ""
2080  IF A$(TR,TC) = "FD2" THEN  XDRAW 34 AT
      TC * 14,Y + (TR * 9):A$(TR,TC) = ""
2090  IF A$(TR,TC) = "" THEN  XDRAW 36 AT TY *
      14,Y + (TX * 9):A$(TR,TC) = A$(I,J):XC(K
      ) = TR:YC(K) = TC:A$(I,J) = "": XDRAW 36
       AT TC * 14,Y + (TR * 9):TX(K) = TR:TY(K
      ) = TC: GOTO 2440
2100  IF A$(TR,TC) = "MI2" THEN  XDRAW 36 AT
      TY * 14,Y + (TX * 9): FOR D = 1 TO 50: XDRAW
      35 AT TC * 14,Y + (TR * 9): NEXT :A$(TR,
      TC) = "":XC(K) = Ø:YC(K) = Ø:A$(TX,TY) =
      "": HOME : PRINT  CHR$ (7)"TANK HIT MINE
       AND WAS DESTROYED!": GOSUB 3640:S = N: RETURN

2110  RETURN
2120  REM  MOVE UP
2130  IF I = 1 THEN 2040
2140 NU = 3: GOSUB 3530: IF I - N < 1 THEN 20
      40
2150  IF MV <  = Ø THEN  RETURN
2160  IF MV - N < Ø THEN 2140
2170 TR = I - N:TC = J:D = 1
2180  GOTO 2060
2190  REM  MOVE DOWN
2200  IF I = 16 THEN 2040
2210 NU = 3: GOSUB 3530: IF I + N > 16 THEN 2
      040
2220  IF MV <  = Ø THEN  RETURN
2230  IF MV - N < Ø THEN 2210
2240 TR = I + N:TC = J:D = 2
2250  GOTO 2060
2260  REM  MOVE LEFT
2270  IF J = 1 THEN 2040
2280 NU = 3: GOSUB 3530: IF J - N < 1 THEN 20
      40
2290  IF MV <  = Ø THEN  RETURN
2300  IF MV - N < Ø THEN 2280
2310 TR = I:TC = J - N:D = 3
2320  GOTO 2060
2330  REM   MOVE RIGHT
2340  IF J = 18 THEN 2040
2350 NU = 3: GOSUB 3530: IF J + N > 18 THEN 2
      040
2360  IF MV <  = Ø THEN  RETURN
2370  IF MV - N < Ø THEN 2350
2380 TR = I:TC = J + N:D = 4
2390  GOTO 2060
2400  IF FL < I THEN N = 1:D = 1: GOTO 2050
2410 N = 2:D = 2: GOTO 2050
2420  IF FL < J THEN N = 3:D = 3: GOTO 2050
2430 N = 4:D = 4: GOTO 2050
2440 S = N: IF N < 3 AND MV - S > Ø THEN 2460
2450  RETURN
2460 NU = 2: GOSUB 3530
2470 I = TX(K):J = TY(K):TX = I:TY = J:XC = I
      :YC = J
2480  IF D <  = 2 THEN  ON N GOSUB 2600,2650:
       RETURN
2490  IF D > 2 THEN  ON N GOSUB 2500,2550: RETURN

2500  IF I = 1 THEN  RETURN
2510  GOSUB 2770: IF N = Ø THEN  RETURN
2520  IF MV - (S + N) < Ø THEN  RETURN
2530  IF I - N < 1 THEN  RETURN
2540 TR = I - N:TC = J: GOTO 2700
2550  IF I = 16 THEN  RETURN
2560  GOSUB 2770: IF N = Ø THEN  RETURN
2570  IF MV - (S + N) < Ø THEN  RETURN
2580  IF I + N > 16 THEN  RETURN
2590 TR = I + N:TC = J: GOTO 2700
2600  IF J = 1 THEN  RETURN
2610  GOSUB 2770: IF N = Ø THEN  RETURN
2620  IF MV - (S + N) < Ø THEN  RETURN
2630  IF J - N < 1 THEN  RETURN
2640 TR = I:TC = J - N: GOTO 2700
2650  IF J = 18 THEN  RETURN
2660  GOSUB 2770: IF N = Ø THEN  RETURN
2670  IF MV - (S + N) < Ø THEN  RETURN
```

continued on page 67

Unlock the Mysteries of your Apple . . . with APPLE SECRETS!

Over 80 of the best Apple Tips and Techniques from Nibble Magazine collected in one soft-cover book! Apple Secrets is packed with programs and information, most of it out of print and unavailable anywhere else. Nibble's Apple experts teach you their tricks for speeding up Applesoft, controlling DOS, creating spectacular graphics, and programming faster and better. And you can get all of the programs from Apple Secrets on disk!

Yes! Send me Apple Secrets! Please send me:

☐ Apple Secrets Book—$19.95
☐ Apple Secrets Book and Disk—$29.95

Please add $1.75 shipping per copy. For shipment outside the U.S., please add $2.75 for surface mail or $6.50 for air mail. (Mass. residents add 5% sales tax) Payable in U.S. funds only.

☐ MasterCard ☐ Visa ☐ Check, M.O.

Name

Address

City State Zip

Signature

Charge Card # Exp. Date

nibble 45 Winthrop St., Concord, MA 01742 (617) 371-1660

LISTING 1: TANK.COMBAT *(continued)*

```
2680  IF J + N > 18 THEN  RETURN
2690 TR = I:TC = J + N
2700 FL = 0
2710  IF A$(TR,TC) = "HQ2" THEN  XDRAW 32 AT
     TC * 14,Y + (TR * 9):A$(TR,TC) = ""
2720  IF A$(TR,TC) = "AD2" THEN  XDRAW 33 AT
     TC * 14,Y + (TR * 9):A$(TR,TC) = ""
2730  IF A$(TR,TC) = "FD2" THEN  XDRAW 34 AT
     TC * 14,Y + (TR * 9):A$(TR,TC) = ""
2740  IF A$(TR,TC) = "" THEN  XDRAW 36 AT TY *
     14,Y + (TX * 9):A$(TR,TC) = A$(I,J):A$(I
     ,J) = "": XDRAW 36 AT TC * 14,Y + (TR *
     9):TX(K) = TR:TY(K) = TC:XC(K) = TR:YC(K
     ) = TC:S = S + N: RETURN
2750  IF A$(TR,TC) = "MI2" THEN  XDRAW 36 AT
     TY * 14,Y + (TX * 9): FOR D = 1 TO 30: XDRAW
     35 AT TC * 14,Y + (TR * 9): NEXT :A$(TR,
     TC) = "":A$(TX,TY) = "":XC(K) = 0:YC(K) =
     0: HOME : PRINT  CHR$ (7)"TANK HIT MINE
     AND WAS DESTROYED!": GOSUB 3640:S = S +
     N: RETURN
2760  RETURN
2770 NU = 3 - S: GOSUB 3530: RETURN
2780  REM  DRAW GRID
2790  HGR : HCOLOR= 3: HPLOT 0,0: CALL  - 308
     2
2800  HCOLOR= 0: FOR I = 8 TO 161 STEP 9: HPLOT
     0,I TO 279,I: NEXT
2810 J = 1
2820  FOR I = 13 TO 265 STEP 14: HPLOT I,0 TO
     I,161
2830  IF J = 19 THEN 2850
2840  DRAW J AT I + 4,7: DRAW J AT I + 4,159:
     J = J + 1
2850  NEXT
2860 J = 1
2870  FOR I = 16 TO 159 STEP 9
2880 A$ =  STR$ (J)
2890  IF  LEN (A$) < 2 THEN A$ = " " + A$
2900  IF  LEFT$ (A$,1) = " " THEN 2920
2910  DRAW ( VAL ( LEFT$ (A$,1))) + 20 AT 1,I
     : DRAW ( VAL ( LEFT$ (A$,1))) + 20 AT 26
     8,I
2920 S =  VAL ( RIGHT$ (A$,1)):S = S + 20: IF
     S = 20 THEN S = 30
2930  DRAW S AT 6,I: DRAW S AT 273,I
2940 J = J + 1: NEXT
2950 Y = 7:J = 1: HCOLOR= 1
2960  FOR I = 1 TO 8: FOR K = 1 TO 18
2970 NU = 10: GOSUB 3530: IF N = 2 THEN A$(I,
     K) = "B": DRAW 20 AT K * 14,Y + (I * 9):
     J = J + 1
2980  IF J = 15 THEN K = 18:I = 8
2990  NEXT K,I
3000  IF J < 15 THEN 2960
3010 J = 1
3020  FOR I = 16 TO 10 STEP  - 1: FOR K = 1 TO
     18
3030  GOSUB 3530: IF N = 2 THEN A$(I,K) = "B"
     : DRAW 20 AT K * 14,Y + (I * 9):J = J +
     1
3040  IF J = 15 THEN K = 18:I = 10
3050  NEXT K,I
3060  IF J < 15 THEN 3020
3070  REM  COMPUTER LOCATIONS
3080  FOR I = 1 TO 8: FOR J = 1 TO 18
3090  GOSUB 3530: IF N = 3 AND A$(I,J) = "" THEN
     A$(I,J) = "HQ": XDRAW 32 AT J * 14,Y + (
     I * 9):I = 8:J = 18:FL = 1
3100  NEXT J,I
3110  IF  NOT FL THEN 3080
3120 FL = 0
3130  FOR I = 1 TO 8: FOR J = 1 TO 18
3140  GOSUB 3530
3150  IF N = 3 AND A$(I,J) = "" THEN A$(I,J) =
     "AD": XDRAW 33 AT J * 14,Y + (I * 9):I =
     8:J = 18:FL = 1
3160  NEXT J,I:: IF  NOT FL THEN 3130
3170 FL = 0
3180  FOR I = 1 TO 8: FOR J = 1 TO 18
3190  GOSUB 3530: IF N = 3 AND A$(I,J) = "" THEN
     A$(I,J) = "FD": XDRAW 34 AT J * 14,Y + (
     I * 9):I = 8:J = 18:FL = 1
3200  NEXT J,I: IF  NOT FL THEN 3180
3210 FL = 0
```

continued on next page

```
3760  IF A$(R,C) = "B" OR A$(R,C) = "HQ2" OR
      A$(R,C) = "AD2" OR A$(R,C) = "FD2" THEN
      3850
3770  IF FL THEN FL = 0: GOTO 3760
3780  IF MT THEN  GOTO 3800
3790  XDRAW SH AT C * 14,Y + (R * 9): IF R =
      TX AND C = TY AND SH > 20 AND SH < 27 THEN
       XDRAW SH AT C * 14,Y + R * 9:MT = 1
3800  POKE 49168,0: WAIT  - 16384,128:A$ =  CHR$
      ( PEEK ( - 16384) - 128): POKE 49168,0
3810  IF MT THEN  IF A$ <  > "I" AND A$ <  >
      "J" AND A$ <  > "K" AND A$ <  > "M" AND
      A$ <  >  CHR$ (32) THEN 3710
3820  IF MT THEN MT = 0: GOTO 3840
3830  XDRAW SH AT C * 14,Y + (R * 9): IF A$ <
       > "I" AND A$ <  > "J" AND A$ <  > "K" AND
      A$ <  > "M" AND A$ <  >  CHR$ (32) THEN
      3710
3840  IF A$ =  CHR$ (32) THEN 4010
3850  IF A$ = "I" THEN 3900
3860  IF A$ = "J" THEN 3930
3870  IF A$ = "K" THEN 3960
3880  IF A$ = "M" THEN 3990
3890  REM  MOVE UP
3900  IF R = RE THEN R = RS: GOTO 3710
3910 R = R - 1: GOTO 3710
3920  REM  MOVE LEFT
3930  IF C = 1 THEN C = 18: GOTO 3710
3940 C = C - 1: GOTO 3710
3950  REM  MOVE RIGHT
3960  IF C = 18 THEN C = 1: GOTO 3710
3970 C = C + 1: GOTO 3710
3980  REM  MOVE DOWN
3990  IF R = RS THEN R = RE: GOTO 3710
4000 R = R + 1: GOTO 3710
4010  IF SH = 37 OR SH > 20 AND SH < 27 THEN
      4030
4020  IF A$(R,C) <  > "" THEN  HOME : PRINT  CHR$
      (7)"THAT SPACE IS OCCUPIED!": FOR D = 1 TO
      2000: NEXT : GOSUB 4120: GOTO 3710
4030  IF SH > 20 AND SH < 27 THEN  RETURN
4040  XDRAW SH AT C * 14,Y + (R * 9): RETURN

4050  IF I = 1 THEN  RETURN
4060  FOR J = 1 TO I
4070  IF R = HX(J) AND C = HY(J) THEN FL = 1
4080  NEXT J
4090  IF FL THEN  POP :FL = 0: GOTO 3850
4100  RETURN
4110  REM  PRINT I-J-K-M PROMPT
4120  HOME : PRINT "ENTER LOCATION OF ";OB$: PRINT
      "USE I-J-K-M TO MOVE": PRINT "<SPACE> TO
       PLACE ";OB$;: RETURN
```

END OF LISTING 1

```
                KEY PERFECT 5.0
                    RUN ON
                TANK.COMBAT
=======================================
   CODE-5.0   LINE# - LINE#   CODE-4.0
   --------   -------------   --------
   3C9E4505     10 -   100      ADD6
   E182930D    110 -   200      A60A
   13834C6F    210 -   300      83BD
   98963C43    310 -   400      F91A
   1727D034    410 -   500    0113FB
   16609335    510 -   600    01A186
   D3D054F0    610 -   700      7761
   8A19BD64    710 -   800      756A
   9A8B880F    810 -   900      82A8
   21461CCB    910 -  1000    012250
   956D3F14   1010 -  1100      C352
   A2C15017   1110 -  1200      4D33
   44490541   1210 -  1300      43D8
   13200042   1310 -  1400      3951
   33F24B56   1410 -  1500    014F4A
   55173D67   1510 -  1600      8CE3
   34660436   1610 -  1700      ACD4
   E360AEFC   1710 -  1800      88D1
   1BF5CC71   1810 -  1900      67A6
   D1CFFD9E   1910 -  2000      5E94
   4D166CAB   2010 -  2100    0121AD
   CC34D15C   2110 -  2200      3C16
   E2742FF9   2210 -  2300      4674
   6C8CBE76   2310 -  2400      44EB
```

continued on next page

```
D0610171    2410 -  2500    5DBD
3A5452C2    2510 -  2600    438C
B1B92ACD    2610 -  2700    3CB1
7F6DAF08    2710 -  2800    0106B7
4C7746BE    2810 -  2900    4226
8B0A8378    2910 -  3000    6CB2
34B59D85    3010 -  3100    68C0
BE57EEB6    3110 -  3200    6FAC
DFA6FFA7    3210 -  3300    7EE9
DFDF4A61    3310 -  3400    843B
D3B34746    3410 -  3500    867F
5DA58092    3510 -  3600    6236
20B76DF7    3610 -  3700    7C45
4ED49F74    3710 -  3800    A947
4B1320DA    3810 -  3900    718D
B807BE46    3910 -  4000    46F9
F5D65C42    4010 -  4100    6126
DE00B952    4110 -  4120    294E
239F7EA2 = PROGRAM TOTAL =  3184
```

LISTING 2: TANK.SHAPES

```
0803- 25 00 4C 00 58
0808- 00 66 00 72 00 7E 00 8B
0810- 00 95 00 A2 00 AF 00 B8
0818- 00 C1 00 CE 00 D6 00 E3
0820- 00 EF 00 FB 00 05 01 12
0828- 01 1F 01 32 01 54 01 5C
0830- 01 68 01 75 01 81 01 8E
0838- 01 9C 01 A5 01 B3 01 C0
0840- 01 CF 01 E3 01 F5 01 06
0848- 02 16 02 2E 02 43 02 21
0850- 24 64 0C 15 15 3E 3F 4E
0858- 31 26 00 21 24 24 2C 2D
0860- 15 F6 3F 4E 31 1E 3F 04
0868- 00 09 2D 05 F8 1B 24 24
0870- 0C 2D 15 04 00 29 2D 05
0878- 20 24 E4 3F 37 36 36 04
0880- 00 29 2D E5 DB 24 2D E5
0888- 1B 24 2D 2D 04 00 21 24
0890- 24 2C 2D B5 1A 3F 04 00
0898- 09 2D 05 20 3C BF 1A 24
08A0- 24 0C 2D 25 00 21 24 24
08A8- AC 12 2D 04 40 36 36 36
08B0- 04 00 09 2D 1C 24 24 07
08B8- 28 25 00 01 A8 2D 20 24
08C0- 3C 28 25 00 21 24 24 6C
08C8- 09 1E 1E 1E 0E 0E 0E 04
08D0- 00 29 2D E5 DB 24 24 24
08D8- 00 21 24 24 AC 0E 2E 80
08E0- 60 36 36 36 04 00 21 24
08E8- 24 AC 72 0E 56 24 24 24
08F0- 04 00 09 2D 05 20 24 E4
08F8- 3F 17 36 36 04 00 21 24
0900- 24 2C 2D 15 F6 3F 04 00
0908- 01 20 24 64 2D 15 36 B6
0910- 07 E0 16 27 00 21 24 24
0918- 2C 2D 15 F6 3F 0E 15 15
0920- 04 00 49 2D 2D 28 20 24
0928- 07 38 38 3F 17 17 36 76
0930- 05 40 25 27 00 29 0D 6D
0938- 6D E5 FB FB FB 63 6D 6D
0940- 6D E5 FB FB FB 63 6D 6D
0948- 6D E5 FB FB FB 63 6D 6D
0950- 6D E5 FB FB FB 23 00 09
0958- 2D 1C 24 24 BC 04 00 29
0960- 2D E5 DB 64 2D 05 20 1C
0968- 3F 27 00 08 15 2D 05 20
0970- 1C 67 21 1C 3F 17 04 00
0978- 49 21 05 38 3F 27 0C 0C
0980- 0C 36 26 00 08 15 2D 05
0988- 20 1C 3F 27 24 2D 2D 04
0990- 00 09 2D 05 20 1C 3F D6
0998- 24 24 0C 2D 15 04 00 09
09A0- 24 0C 0C 0C 3C 3F 27 00
09A8- 09 2D 05 20 1C 0C E4 3F
09B0- 17 6E 3A 17 26 00 08 0E
09B8- 2D 05 20 24 E4 3F 17 76
09C0- 2D 04 00 09 2D 05 20 24
09C8- E4 3F 17 36 36 2D 20 64
09D0- 04 00 49 24 24 35 36 6E
09D8- 09 25 24 3C 36 3E 3F 2C
09E0- 25 3F 0C 24 04 00 09 80
09E8- 20 24 AC 2A 44 36 36 0D
09F0- 20 64 AD B6 1F 0C 04 00
09F8- 09 20 24 0C AD 9F 2D 36
0A00- 0D 24 24 2D 15 36 1E 27
0A08- 00 09 20 24 2C 2D DE 2A
0A10- 4D 24 2D 15 36 1E 3F 24
0A18- 00 69 09 4D E1 DF DF 0C
0A20- 0D 0D 1C 3F 2C 2D F8 1F
0A28- 07 68 69 29 F8 1B DF 23
0A30- 00 49 49 20 05 38 27 0D
0A38- 3C 3F 36 3E 24 24 6D 09
0A40- 35 36 3E 24 24 00 24 24
0A48- 24 2C 2D 2D 2D 2D 2D 35
0A50- 36 36 36 3F 3F 3F 3F 3F
0A58- 27 24 24 2C 2D 2D 2D 2D
0A60- 35 36 36 3F 3F 3F 3F 27
0A68- 24 2C 2D 2D 2D 35 36 3F
0A70- 3F 3F 27 2C 2D 2D 35 3F
0A78- 3F 27 00
```

END OF LISTING 2

PROCURSOR

TIPS 'N TECHNIQUES

by Jim Lazar

If you're yearning for the old solid box cursor on your //e or //c, now you can have it with this ProDOS routine. You can even choose a cursor character of your own.

Return with us now to the days of the blinking box cursor. When I bought my Apple //e, I was disappointed by the lack of the old, solid, blinking box cursor. By the time I upgraded to ProDOS, I was starved for the friendly, blinking box. To finally rid myself of the checkerboard cursor and the 80-column non-blinking cursor, I wrote ProCursor.

ProCursor replaces your system's cursor with the solid, blinking cursor that any veteran user of the Apple II or II Plus will recognize. ProCursor features include speedy movement on the 80-column screen, and the ability to convert the cursor to any character you want (such as an underline or even an apple-shaped character on the Apple //c).

ProCursor works on any Apple //c or //e under ProDOS with BASIC.SYSTEM installed. It operates exactly as the normal cursor does and is compatible with all Applesoft programs and any machine language programs that use the Monitor's KEYIN routine.

USING PROCURSOR

To install ProCursor, either execute the machine language program (**Listing 1**) directly or run the Applesoft program (**Listing 2**). There are three ways to execute the machine language program directly:

BRUN PROCURSOR

or:

-PROCURSOR

or, from within a program:

PRINT CHR$(4)"-PROCURSOR"

To run the Applesoft program (**Listing 2**), type:

RUN PROCURSOR.INST

The program automatically determines what type of //e or //c you have, and then offers you a choice of two cursor characters:

1. Solid Blinking Box
2. Custom Character

If you choose option 2, you are first asked for the code of your custom character. Consult **Table 1** for some commonly used character codes.

Once the program is installed, it should

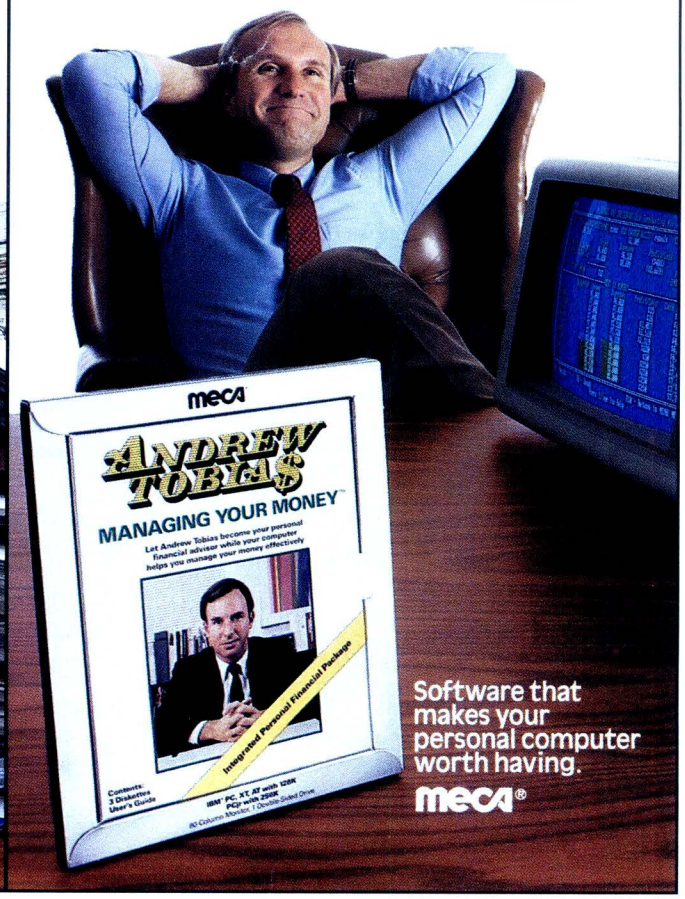
TABLE 1: Blink Character Values

Hex	Decimal	Blink Character
$FF	255	Normal checkerboard
$DF/$9F	223/159	Normal underline
$DE/$9E	222/158	Normal caret (^)
$AA	170	Normal asterisk (*)
$A0	160	Normal space
$7F	127	Inverse checkerboard
$2B	43	Inverse plus sign (+)
$20	32	Inverse space
$1E	30	Inverse caret (^)
	Apple //c and Enhanced Apple //e Only	
$40	64	Solid apple
$41	65	Open apple
$42	66	Pointer
$5B	91	Diamond
$5D	93	Cross

by Jim Lazar, 1109 Niesen Rd., Port Washington, WI 53074. ProCursor is compatible with ProDOS only.

MANAGING YOUR CHECKBOOK?
MANAGING YOUR BUDGET?
MANAGING YOUR BILLS?
MANAGING YOUR CASH FLOW?
MANAGING YOUR TAXES?
MANAGING YOUR INSURANCE?
MANAGING YOUR STOCKS?
MANAGING YOUR BONDS?
MANAGING YOUR REAL ESTATE?
MANAGING YOUR TAX SHELTERS?
MANAGING YOUR SAVINGS?
MANAGING YOUR MORTGAGE?
MANAGING YOUR AUTO LOAN?
MANAGING YOUR RETIREMENT?
MANAGING YOUR CALENDAR?
MANAGING YOUR CHARGE ACCOUNTS?
MANAGING YOUR CAPITAL GAINS?
MANAGING YOUR ANNUITIES?
MANAGING YOUR APPOINTMENTS?
MANAGING YOUR DIVIDENDS?
MANAGING YOUR INTEREST?
MANAGING YOUR RECORDS?
MANAGING YOUR VALUABLES?
MANAGING YOUR KEOGH'S?
MANAGING YOUR IRA'S?

MANAGING YOUR MONEY®
BY ANDREW TOBIAS.
THE ONLY FINANCIAL
SOFTWARE THAT DOES IT ALL.



Software that
makes your
personal computer
worth having.

meca®

Nibble Light Pen, ProDOS Directory List, ProCursor, DISPLAY and programs from Nibbling at Assembly Language V are available on diskette for an introductory price of $17.95 plus $1.50 shipping/handling ($2.50 outside the U.S.) from Nibble, 45 Winthrop St., Concord, MA 01742. Introductory price expires 3/31/86.

LISTING 1: PROCURSOR

```
0000:                    1 ************************
0000:                    2 *      PROCURSOR       *
0000:                    3 *    BY JIM LAZAR      *
0000:                    4 * COPYRIGHT (C) 1986 *
0000:                    5 * BY MICROSPARC, INC *
0000:                    6 * CONCORD, MA  01742 *
0000:                    7 ************************
0000:                    8 *
0000:                    9 *  Tool Kit Assembler
0000:                   10 *
0000:                   11 *  Equates
0000:                   12 *

0000:                   15 ************************************************************
0000:                   16 *  Zero Page                                               *
0000:                   17 ************************************************************
0000:          0024     18 HCUR       EQU   $24
0000:          0028     19 BAS        EQU   $28
0000:          004E     20 RNDL       EQU   $4E
0000:          004F     21 RNDH       EQU   $4F
0000:          00FB     22 WAITLEN    EQU   $FB
0000:          00FC     23 CHAR       EQU   $FC
0000:          00FD     24 OLDCHAR    EQU   $FD
0000:          00FE     25 SPECIAL    EQU   $FE
0000:          00FF     26 BNKCHAR    EQU   $FF
0000:                   27 ************************************************************
0000:                   28 *  80-Column Card                                          *
0000:                   29 ************************************************************
0000:          057B     30 HCUR80     EQU   $57B
0000:                   31 ************************************************************
0000:                   32 *  Basic System                                            *
0000:                   33 ************************************************************
0000:          BE36     34 DOSIN      EQU   $BE36
0000:          BE44     35 IFILACTV   EQU   $BE44
0000:          BE43     36 EXACTV     EQU   $BE43
0000:                   37 ************************************************************
0000:                   38 *  Hardware                                                *
0000:                   39 ************************************************************
0000:          C000     40 KEYDATA    EQU   $C000
0000:          C00F     41 ALTSET     EQU   $C00F
0000:          C01F     42 COL80      EQU   $C01F
0000:          C054     43 TEXT1      EQU   $C054
0000:          C055     44 TEXT2      EQU   $C055

0000:                   46 ************************************************************
0000:                   47 *  ProCursor                                               *
0000:                   48 ************************************************************
----- NEXT OBJECT FILE NAME IS PROCURSOR
02E9:          02E9     49            ORG   $2E9
02E9:AD 36 BE           50            LDA   DOSIN          ;Set up Basic.System
02EC:8D 83 03           51            STA   EXITVECT       ; Global Page vectors
02EF:AD 37 BE           52            LDA   DOSIN+1        ; and exit address
02F2:8D 84 03           53            STA   EXITVECT+1
02F5:A9 00              54            LDA   #>PROCUR
02F7:8D 36 BE           55            STA   DOSIN
02FA:A9 03              56            LDA   #<PROCUR
02FC:8D 37 BE           57            STA   DOSIN+1
02FF:60                 58            RTS

0300:                   60 ************************************************************
0300:                   61 *  Cursor Flashing Routine                                 *
0300:                   62 ************************************************************
0300:2C 44 BE           63 PROCUR     BIT   IFILACTV       ;Check to see if text
0303:30 05   030A       64            BMI   JMPEXIT        ; file is being input
0305:2C 43 BE           65            BIT   EXACTV         ; or EXEC file active
0308:10 03   030D       66            BPL   FLASH          ; and skip ProCursor if
030A:4C 82 03           67 JMPEXIT    JMP   EXIT           ; either is
030D:48                 68 FLASH      PHA
030E:85 FC              69            STA   CHAR           ;Save original character
0310:8A                 70            TXA                   ; and registers
0311:48                 71            PHA
0312:98                 72            TYA
0313:48                 73            PHA
0314:A9 00              74            LDA   #$00           ;Clear special character
0316:85 FE              75            STA   SPECIAL        ; flag
0318:2C 1F C0           76            BIT   COL80          ;If 80-Column Card is on
031B:10 1F   033C       77            BPL   CHKCHAR        ; then clean up the
031D:A4 24              78            LDY   HCUR           ; Monitor's attempt
031F:A5 FC              79            LDA   CHAR           ; to flash the screen
0321:91 28              80            STA   (BAS),Y        ; character
0323:AD 7B 05           81            LDA   HCUR80         ;Divide horizontal
0326:4A                 82            LSR                  ; cursor position by two
0327:A8                 83            TAY
0328:08                 84            PHP                   ;Lock interrupts while
0329:78                 85            SEI                   ; screenholes are wrong
032A:AD 55 C0           86            LDA   TEXT2          ;Assume even column
032D:90 03   0332       87            BCC   GETOLD
032F:AD 54 C0           88            LDA   TEXT1          ;Nope it's an odd column
0332:B1 28              89 GETOLD     LDA   (BAS),Y        ;Pick up the character
0334:85 FC              90            STA   CHAR           ; and save it
0336:85 FD              91            STA   OLDCHAR
0338:AD 54 C0           92            LDA   TEXT1          ;Restore to page 1
033B:28                 93            PLP

033C:                   95 ************************************************************
033C:                   96 *  Check original character for special values             *
033C:                   97 ************************************************************
```

continued on next page

not be installed again unless you have disabled it or rebooted. To disable PROCURSOR, enter the following two POKEs:

POKE 48694,PEEK(899): POKE 48695, PEEK(900)

ENTERING THE PROGRAM

With an Assembler

If you have an assembler, enter the program as shown in **Listing 1**. PROCURSOR produces a solid block cursor on the enhanced //e or //c. To produce a version that displays a solid block cursor on the original //e, enter **Listing 1** as shown and save the object code. Then perform the following steps to install a patch:

1. BLOAD PROCURSOR
2. CALL −151
3. Enter the patch: **33C:4C 58 03**
4. BSAVE PROCURSOR,A$2E9,L$DA

To save PROCURSOR with a custom cursor, determine the character code for the cursor you want to use, assemble **Listing 1**, and follow the patch procedure described above, substituting the following patch:

33C: A9 80 85 FE A9 *code* 85 FF 4C 58 03

where *code* is the ASCII code of the new cursor character in hexadecimal notation.

If you have entered PROCURSOR using an assembler, you do not need to enter **Listing 2**.

Without an Assembler

If you don't have an assembler, do not try to enter **Listing 1**. Instead, enter the Applesoft program in **Listing 2** and save it on disk with the command:

SAVE PROCURSOR.INST

After you run this program and select the cursor you want, it will automatically create the machine language program. When the program ends, you may save the resulting machine language program with the command:

BSAVE PROCURSOR,A$2E9,L$DA

For help in entering *Nibble* listings, see "A Welcome to New *Nibble* Readers" at the beginning of this issue.

HOW IT WORKS

My first thought in writing ProCursor was to tap into the BASIC interpreter's input/output structure. Unfortunately, there is no one area in the BASIC interpreter (BI) where a patch could be made that would remain valid after any revision of the BI.

Finally, I came up with the best method to flash the cursor: any request for a keypress (through the Monitor keyboard input vector) is directed to PROCURSOR instead of to the BI's keyboard intercept routine. (In other words, we're intercepting the request for a keypress twice.) PROCURSOR flashes

LISTING 1: PROCURSOR *(continued)*

```
033C:A5 FC          98 CHKCHAR   LDA   CHAR          ;Check original character
033E:C9 E0          99           CMP   #$E0         ; for the values $C0-$DF
0340:B0 16   0358  100           BCS   GETKEY       ; and change them to
0342:C9 C0         101           CMP   #$C0         ; $80-$9F to avoid //c
0344:90 04   034A  102           BCC   CHKCHAR1     ; special characters
0346:E9 40         103           SBC   #$40
0348:85 FC         104           STA   CHAR
034A:C9 60         105 CHKCHAR1  CMP   #$60         ;Check for a //c special
034C:B0 0A   0358  106           BCS   GETKEY       ; character and set flag
034E:C9 40         107           CMP   #$40         ; and save a space as the
0350:90 06   0358  108           BCC   GETKEY       ; negative cursor image
0352:A9 A0         109           LDA   #$A0
0354:85 FE         110           STA   SPECIAL
0356:85 FF         111           STA   BNKCHAR

0358:              113 ***************************************************
0358:              114 *  Get keypress while flashing cursor             *
0358:              115 ***************************************************
0358:20 85 03      116 GETKEY    JSR   BLINK        ;Blink cursor
035B:E6 4E         117           INC   RNDL         ;Increment random number
035D:D0 02   0361  118           BNE   GETKEY1      ; for monitor
035F:E6 4F         119           INC   RNDH
0361:A2 FF         120 GETKEY1   LDX   #$FF
0363:86 FB         121           STX   WAITLEN
0365:2C 00 C0      122 KEYWAIT1  BIT   KEYDATA      ;Check for keypress
0368:30 09   0373  123           BMI   KEYFND       ;Yes, key was pressed
036A:CA            124 KEYWAIT2  DEX                 ;Wait for a bit then
036B:D0 FD   036A  125           BNE   KEYWAIT2     ; check key again
036D:C6 FB         126           DEC   WAITLEN
036F:D0 F4   0365  127           BNE   KEYWAIT1     ;Okay waited long enough
0371:F0 E5   0358  128           BEQ   GETKEY       ; blink cursor again
0373:2C 1F C0      129 KEYFND    BIT   COL80        ;If 80-Column Card is on
0376:10 05   037D  130           BPL   KEYFNDX      ; then repair the screen
0378:A5 FD         131           LDA   OLDCHAR
037A:20 AB 03      132           JSR   STORE
037D:68            133 KEYFNDX   PLA                 ;Restore all registers
037E:A8            134           TAY                 ; before calling real
037F:68            135           PLA                 ; input routine
0380:AA            136           TAX
0381:68            137           PLA

0382:              139 ***************************************************
0382:              140 *  Exit vector                                    *
0382:              141 ***************************************************
0382:4C            142 EXIT      DFB   $4C          ;Call real input routine
0383:00 00         143 EXITVECT  DW    $0000

0385:              145 ***************************************************
0385:              146 *  Blink Cursor                                   *
0385:              147 ***************************************************
0385:8D 0F C0      148 BLINK     STA   ALTSET       ;Blink cursor
0388:24 FE         149           BIT   SPECIAL      ;If a special character
038A:10 0B   0397  150           BPL   BLINK1       ; is on screen
038C:A6 FF         151           LDX   BNKCHAR      ; then switch character
038E:A5 FC         152           LDA   CHAR         ; with negative cursor
0390:85 FF         153           STA   BNKCHAR      ; character
0392:8A            154           TXA
0393:85 FC         155           STA   CHAR
0395:D0 06   039D  156           BNE   SAVE
0397:A5 FC         157 BLINK1    LDA   CHAR         ;Make character opposite
0399:49 80         158           EOR   #$80         ; mode
039B:85 FC         159           STA   CHAR
039D:2C 1F C0      160 SAVE      BIT   COL80        ;Save cursor on screen
03A0:30 07   03A9  161           BMI   SAVE80
03A2:A4 24         162           LDY   HCUR         ;Save on 40-column screen
03A4:A5 FC         163           LDA   CHAR
03A6:91 28         164           STA   (BAS),Y
03A8:60            165           RTS
03A9:A5 FC         166 SAVE80    LDA   CHAR         ;Save on 80-column screen

03AB:              168 ***************************************************
03AB:              169 *  Store character on screen                      *
03AB:              170 ***************************************************
03AB:48            171 STORE     PHA
03AC:AD 7B 05      172           LDA   HCUR80       ;Divide horizontal
03AF:4A            173           LSR                ; cursor position by two
03B0:A8            174           TAY
03B1:68            175           PLA
03B2:08            176           PHP
03B3:78            177           SEI                 ;Set interrupt disable
03B4:2C 55 C0      178           BIT   TEXT2        ;Assume even column
03B7:90 03   03BC  179           BCC   STORE1
03B9:2C 54 C0      180           BIT   TEXT1        ;Nope-odd column
03BC:91 28         181 STORE1    STA   (BAS),Y      ;Store onto the screen
03BE:2C 54 C0      182           BIT   TEXT1
03C1:28            183           PLP
03C2:60            184           RTS
```

END OF LISTING 1

```
KEY PERFECT 5.0
RUN ON
PROCURSOR
=====================================
CODE-5.0   ADDR# - ADDR#   CODE-4.0
--------   -------------   --------
489ED784     02E9 - 0338     29BE
6536602A     0339 - 0388     2909
9D6BC6E3     0389 - 03C2     2166
7B353FCB = PROGRAM TOTAL =     DA
```

cursor (as long as an EXEC file isn't in effect or a text file isn't being read from) and jumps to the original intercept routine (in the BI).

When PROCURSOR is run, **lines 50-53** in **Listing 1** save the original keyboard interrupt routine address (at $383) for branching after each keypress. **Lines 54-58** set the keyboard interrupt to point to the beginning of PROCURSOR (the Monitor's vector at $28 is set by the BI).

When PROCURSOR is installed, a jump is made from any routine that uses the standard keyboard input routines to the beginning of PROCURSOR (**line 63**). From here, **lines 63-67** determine if a text file is being read or if an EXEC file is active. The program exits to the normal input routine if either one is active. Next, **lines 68-76** save the processor registers and the original screen character (also, the special flag is cleared).

Lines 76-77 check to see if the 80-column screen is on; if not, program flow skips to **line 98**. If the screen is on, **lines 78-80** repair the Monitor's attempt to flash the character on the screen. **Lines 81-93** pick up the correct character from the screen and save it in CHAR and OLDCHAR.

Lines 98-104 check the original character for a value between $C0 and $DF and change the character to eliminate Mousetext on the Apple //c (values $40-$5F) on the inverse blink. **Lines 105-111** check for a Mousetext character, set the SPECIAL flag and set the blink character to a normal space character ($A0).

If you elected to install the original //e patch or the custom cursor patch, the code at $3CC will be different. In the case of the original //e, the first instruction will be a JMP $358, which skips the check for Mousetext characters. The custom cursor patch makes the program think that the original character was a Mousetext character and loads the Accumulator with the code for the custom character before jumping to $358.

ProCursor replaces your system's cursor with the solid, blinking cursor that any veteran user of the Apple II or II Plus will recognize.

If a key was pressed, **lines 129-132** restore the original character to the 80-column screen. **Lines 133-137** restore the registers before calling the real input routine. The exit vector (**lines 142-143**) jumps to the address that was in memory locations $BE36 and $BE37 when PROCURSOR was installed.

The cursor blinking routine is in **lines 148-166**. **Lines 148-156** swap the original character and the blink character, and save the result on the screen if the special flag is set. **Lines 157-159** change the high bit on the character to switch it from normal to inverse or vice versa. **Lines 160-166** save the new cursor character on the screen. Finally, **lines 171-184** make up the 80-column character storage routine.

CUSTOMIZING THE CURSOR

When using PROCURSOR with the patch to blink the original character with a character of your choice, you can change the blink character by entering:

POKE 833, *code*

where *code* is the character code for your new cursor. This allows you to use different cursors for different inputs. For instance, you could use an underline for strings and a plus sign for numbers.

PROBLEMS WITH PROCURSOR

There are two minor problems with ProCursor that can't be fixed without altering ProDOS, the BASIC interpreter or the 80-column firmware. If you leave escape mode when the 80-column firmware is in effect, the cursor reverts to the checkerboard until the next keypress. Also, if you use the right arrow key to trace over a Mousetext character on the //c (but not the enhanced //e), it turns into an inverse upper-case character, and vice versa.

LISTING 2: PROCURSOR.INST

```
10  REM ************************
20  REM *    PROCURSOR.INST     *
30  REM *      BY JIM LAZAR     *
40  REM * COPYRIGHT (C) 1986 *
50  REM * BY MICROSPARC, INC *
60  REM * CONCORD, MA  01742 *
70  REM ************************
80  REM
90  IF  PEEK (64435) <  > 6 THEN  HOME : VTAB
    12: PRINT "THIS IS NOT A //e OR A //c.":
     END
100  IF  PEEK (48694) = 0 AND  PEEK (48695) =
    3 THEN  POKE 48694, PEEK (899): POKE 486
    95, PEEK (900)
110  HOME : PRINT "ProCursor": PRINT : PRINT
    "By Jim Lazar": PRINT : PRINT : PRINT "C
    opyright 1986": PRINT "by MicroSPARC, I
    nc.": PRINT : PRINT : PRINT
120  PRINT "1-Solid Blinking Box": PRINT "2-C
    ustom Character": PRINT
130  PRINT : INPUT "Enter version to install:
     ";A$: IF  VAL (A$) < 1 OR  VAL (A$) > 2
     THEN  PRINT  CHR$ (7);: GOTO 130
140 VER = 3 * (A$ = "2") + (A$ = "1") * (2 -
    ( PEEK (64448) = 234)): IF VER <  > 3 THEN
    170
150  PRINT : INPUT "Enter blink character num
    ber: ";A$: IF  VAL (A$) < 0 OR  VAL (A$)
     > 255 THEN  PRINT  CHR$ (7);: GOTO 150
160 CHAR =  VAL (A$)
170  FOR A = 745 TO 962: READ B: POKE A,B: NEXT
180  IF VER = 2 THEN  POKE 828,76: POKE 829,8
    8: POKE 830,3: FOR A = 831 TO 855: POKE
    A,234: NEXT : GOTO 210
190  IF VER = 1 THEN 210
200  FOR A = 828 TO 838: READ B: POKE A,B: NEXT
    : POKE 833,CHAR: FOR A = 839 TO 855: POKE
    A,234: NEXT
210  CALL 745: HOME : PRINT "ProCursor 1."VER
    " is installed"
220  DATA 173,54,190,141,131,3,173,55,190,141
    ,132,3,169,0,141,54,190,169,3,141,55,190
    ,96,44,68,190,48,5,44,67,190,16,3,76,130
    ,3,72,133,252,138,72,152,72,169,0,133,25
    4,44,31,192,16,31,164,36,165,252,145,40
230  DATA 173,123,5,74,168,8,120,173,85,192,1
    44,3,173,84,192,177,40,133,252,133,253,1
    73,84,192,40,165,252,201,224,176,22,201,
    192,144,4,233,64,133,252,201,96,176,10,2
    01,64,144,6,169,160,133,254,133,255,32,1
    33,3,230,78
240  DATA 208,2,230,79,162,255,134,251,44,0,1
    92,48,9,202,208,253,198,251,208,244,240,
    229,44,31,192,16,5,165,253,32,171,3,104,
    168,104,170,104,76,0,0,141,15,192,36,254
    ,16,11,166,255,165,252,133,255,138,133
250  DATA 252,208,6,165,252,73,128,133,252,44
    ,31,192,48,7,164,36,165,252,145,40,96,16
    5,252,72,173,123,5,74,168,104,8,120,44,8
    5,192,144,3,44,84,192,145,40,44,84,192,4
    0,96
260  DATA 169,128,133,254,169,0,133,255,76,88
    ,3
```

END OF LISTING 2

DISPLAY
A ProDOS VAR File Reader

by Ken Manly

APPLE UTILITIES

The ProDOS VAR

type file has many uses, including saving memory, increasing speed, and helping with program debugging. Add the DISPLAY command to ProDOS and you can view the contents of any VAR file.

Of the new features that ProDOS brings to Applesoft programming, the VAR file is one of the most useful. VAR files are disk files that contain the names and values of all real, integer and string variables in an Applesoft program. The STORE command creates the VAR file on disk. The RESTORE command loads the variables and their values from disk and substitutes them for any current Applesoft variables.

However, without a program, there is no easy way to see the contents of a VAR file. With DISPLAY, a new command is added to ProDOS that lists the contents of a VAR file.

There are many potential uses for VAR files. Let's consider three: First, a VAR file is a fast and memory-efficient way to initialize program variables. To set up an initialization file, type CLEAR, then type each of the variable definitions you want in your program. Save it with the STORE command followed by your file name. For example, STORE SETUP. Then include the instruction:

```
PRINT CHR$(4)"RESTORE SETUP"
```

at the beginning of your program. This method works faster than having your program read values from text files and it requires less memory than multiple assignment statements or data statements.

Second, a VAR file is a good way to preserve the current state of a program while it carries out another task. Suppose that you occasionally want your program to sort using a very large array. Having the array defined all the time is a waste of precious memory. Using VAR files, you can STORE the current variables, CLEAR memory, define and use your large array, and RESTORE the current variables. When you use the RESTORE command, the space occupied by the large array is cleared, the current variables are restored, and your program goes back to where it left off. If you have an Apple //c or a //e with an extended 80-column card, you can STORE to and RESTORE from the /RAM disk emulator. Storage to /RAM is so quick that the delay may not be noticeable.

Third, VAR files can be useful in program debugging. You can get a snapshot of what the variables look like at any point in your program by inserting a statement such as:

```
PRINT CHR$(4)"STORE VARBLS"
```

If some of the variables in this VAR file do not have their expected values, you have a good idea of where the bug is hiding.

DISPLAY

The third use raises a question: How can you examine the contents of a VAR file? You DISPLAY it, of course. To add the DISPLAY command to ProDOS, just BRUN DISPLAY (**Listing 1**).

The DISPLAY command can be used with slot and drive parameters:

DISPLAY *filename*,[S*slot#*,D*drive#*]

or it can be used with a ProDOS pathname:

DISPLAY *pathname*

Examples of each are:

```
DISPLAY VARBLS,S6,D2
DISPLAY /MYDISK/VARBLS
```

Like other ProDOS commands, DISPLAY can be used from within a program in a PRINT statement preceded by a <CTRL>D as in:

```
PRINT CHR$(4)"DISPLAY SETUP"
```

The specified file must be a VAR type file.

A sample DISPLAY output is shown in **Example 1**. Control characters are indicated by a normal character enclosed in square brackets. Any strings of zero length are identified by "Empty string." Simple variables are displayed first, followed by arrays, both in the order in which they were defined. Array elements are listed in order, starting with the zero element. In listing multidimensional array elements, the first index is the least significant and the last index is the most significant. For example, if BX( ) is dimensioned as BX(1,1), its elements would be listed in the order: BX(0,0), BX(1,0),

Ken Manly, Buffalo Chip Software, 35 Tillinghast Pl., Buffalo, NY 14216. *DISPLAY is compatible with ProDOS only.*

EXAMPLE 1: Sample DISPLAY Output

```
D $ = [D]
AB  = 25.72
TL% = 3
DT$ = JULY 4, 1984
TY  = Function definition
X   = Ø
I   = 6
J   = 3
NL$ = Empty string
AR(2,2) =
Ø
1
4
.1
1.1
4.1
.2
1.2
4.2

ST$(5) =
JU
JULY
JULY 4
JULY 4,
JULY 4, 19
JULY 4, 1984
```

BX(0,1), and BX(1,1).

An array listing can be very long, and you may not need to display all of it. Pressing <ESC> or <CTRL>C during an array listing will cause DISPLAY to stop listing that array and begin listing the next array. Another way to minimize array listing length is to explicitly dimension arrays that are smaller than the default dimension of ten.

The speed of the display can be controlled by the Applesoft SPEED command, and the display can be sent to a printer or other output device with the ProDOS PR#*n* command. DISPLAY is compatible with other ProDOS add-on commands that use the recommended installation methods.

Some of DISPLAY's screen instructions and error messages contain lower-case letters, but if DISPLAY is run on an Apple II Plus, lower-case letters are converted to upper-case as they are printed. Lower-case letters in a VAR file will not be converted, however, on the assumption that the Apple II Plus must have been modified to allow display of lower-case.

USING DISPLAY

Let's look at an example of using DISPLAY. Suppose you have a program that uses text files often. Some of your ProDOS commands are assigned to string variables; for example, D$=CHR$(4), OP$=D$+"OPEN", and CL$=D$+"CLOSE". These are stored in a VAR file called VBLS. The program statement:

PRINT CHR$(4)"RESTORE VBLS"

loads all the variables, but when you go back

Nibble Light Pen, ProDOS Directory List, ProCursor, DISPLAY and programs from Nibbling at Assembly Language V are available on diskette for an introductory price of $17.95 plus $1.50 shipping/handling ($2.50 outside the U.S.) from Nibble, 45 Winthrop St., Concord, MA 01742. Introductory price expires 3/31/86.

LISTING 1: DISPLAY

```
0000:                    1           LST   ON,G
0000:                    2 ********************************
0000:                    3 *                              *
0000:                    4 *           DISPLAY            *
0000:                    5 *        by Ken Manly          *
0000:                    6 *    Buffalo Chip Software     *
0000:                    7 *                              *
0000:                    8 *     Copyright (C)  1986      *
0000:                    9 *     by MicroSPARC, Inc.      *
0000:                   10 *     Concord, MA   01742      *
0000:                   11 *                              *
0000:                   12 *         EDASM.SYSTEM         *
0000:                   13 *                              *
0000:                   14 ********************************
0000:                   15 ;
0000:          002F     16 LENGTH    EQU   $2F
0000:          003C     17 A1L       EQU   $3C
0000:          003A     18 PCL       EQU   $3A
0000:          003E     19 A2L       EQU   $3E
0000:          0042     20 A4L       EQU   $42
0000:          005E     21 INDEX     EQU   $5E
0000:          0069     22 VARTAB    EQU   $69
0000:          0073     23 HIMEM     EQU   $73
0000:          00FC     24 BUFPTR    EQU   $FC
0000:          00FE     25 PRTPTR    EQU   $FE
0000:          0100     26 STACK     EQU   $100
0000:          0200     27 INPUT     EQU   $200
0000:          BE06     28 EXTRNCMD  EQU   $BE06
0000:          BE0C     29 PRINTERR  EQU   $BE0C
0000:          BE50     30 XTRNADDR  EQU   $BE50
0000:          BE52     31 XLEN      EQU   $BE52
0000:          BE53     32 XCNUM     EQU   $BE53
0000:          BE54     33 PBITS     EQU   $BE54
0000:          BE6C     34 VPATH1    EQU   $BE6C
0000:          BE70     35 GOSYSTEM  EQU   $BE70
0000:          BE9E     36 XRETURN   EQU   $BE9E
0000:          BEB4     37 SSGINFO   EQU   $BEB4
0000:          BEB8     38 FIFILID   EQU   $BEB8
0000:          BEB9     39 FIAUXID   EQU   $BEB9
0000:          BEBC     40 FIBLOKS   EQU   $BEBC
0000:          BECE     41 OSYSBUF   EQU   $BECE
0000:          BED0     42 OREFNUM   EQU   $BED0
0000:          BED6     43 RWREFNUM  EQU   $BED6
0000:          BED7     44 RWDATA    EQU   $BED7
0000:          BED9     45 RWCOUNT   EQU   $BED9
0000:          BEDE     46 CREFNUM   EQU   $BEDE
0000:          BEF5     47 GETBUFR   EQU   $BEF5
0000:          BF58     48 BITMAP    EQU   $BF58
0000:          BF98     49 MACHID    EQU   $BF98
0000:          C000     50 KBD       EQU   $C000
0000:          C010     51 KBSTROBE  EQU   $C010
0000:          C030     52 CLICK     EQU   $C030
0000:          DAFB     53 CRDO      EQU   $DAFB
0000:          DB5C     54 OUTDO     EQU   $DB5C
0000:          DEF3     55 INTOFAC1  EQU   $DEF3
0000:          ED24     56 LINPRT    EQU   $ED24
0000:          EAF9     57 MOVFM     EQU   $EAF9
0000:          ED2E     58 PRNTFAC   EQU   $ED2E
0000:          F88C     59 INSDS2    EQU   $F88C
0000:          F953     60 PCADJ     EQU   $F953
0000:          FCA8     61 WAIT      EQU   $FCA8
0000:          FD75     62 NXTCHAR   EQU   $FD75
0000:          FE2C     63 MOVE      EQU   $FE2C
0000:                   64 ;
0000:                   65 ;ProDOS MLI function call codes
0000:                   66 ;
0000:          00CA     67 READ      EQU   $CA
0000:          00C4     68 GETINFO   EQU   $C4
0000:          00C8     69 OPEN      EQU   $C8
0000:          00CC     70 CLOSE     EQU   $CC
0000:                   71 ;
0000:                   72           MSB   ON
----- NEXT OBJECT FILE NAME IS DISPLAY
2000:          2000     73           ORG   $2000
2000:                   74 ;
2000:A9 04              75           LDA   #<LAST-BEGIN+$100 ;Ask BASIC to
2002:20 F5 BE           76           JSR   GETBUFR           ; reserve space
2005:8D 3A 21           77           STA   ORIG              ;Remember where it is
2008:85 3C              78           STA   A1L
200A:38                 79           SEC
200B:E9 22              80           SBC   #<BEGIN           ;Subtract present location
200D:8D 3B 21           81           STA   RELO              ; to see how far program
2010:A9 04              82           LDA   #<LAST-BEGIN+$100 ; will have to be
2012:85 3D              83           STA   A1L+1             ; moved
2014:A5 3C              84 PROTECT   LDA   A1L               ;Protect program space
```

continued on page 79

to work on the program, you cannot remember whether you included the append command (AP$=D$+"APPEND") in your file. To find out, BRUN DISPLAY to install the command. Successful installation is indicated by the title and copyright notice. Now type:

DISPLAY VBLS

You should see something like this:

```
D $ = [D]
OP$ = [D]OPEN
CL$ = [D]CLOSE
RD$ = [D]READ
WR$ = [D]WRITE
```

The APPEND command is not defined, so add it by typing three commands:

RESTORE VBLS
AP$=D$+"APPEND"
STORE VBLS

This procedure is simple if you have a very small VAR file. What if you have defined too many variables to fit on the screen, and they scroll off the screen before you can read through them? There are three things you can do. First, you can use <CTRL>S to make the listing pause. Type <CTRL>S again to continue. Second, you can type **SPEED=100** before you type the DISPLAY command. This will slow down the display so you can read everything as it scrolls by. Third, you can print out your display by typing PR#*n* (where *n* is the slot number of your printer interface) before the DISPLAY command.

ENTERING DISPLAY

Use the Monitor to type in the DISPLAY program from the assembly language listing (**Listing 1**). Save it with the command:

BSAVE DISPLAY,A$2000,L$5CB

If you enter the program with an assembler, note that there is one convention used by Apple's EDASM.SYSTEM that is exactly the opposite in assemblers such as Big Mac or Merlin. In EDASM.SYSTEM, the statement #<EXP refers to the high byte of the two-byte value EXP, and #>EXP refers to the low byte.

For help in entering *Nibble* listings, see "A Welcome to New *Nibble* Readers" at the beginning of this issue.

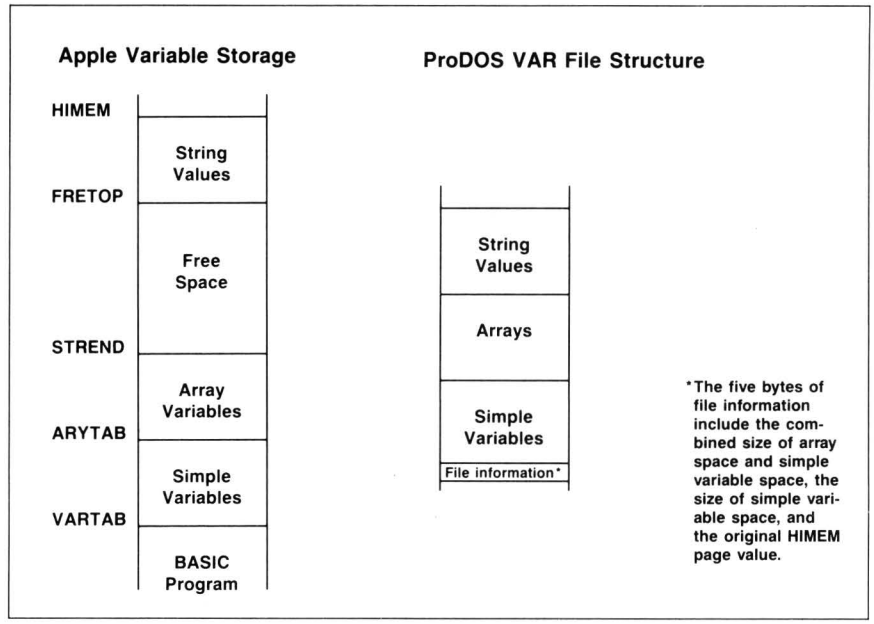
ANATOMY OF A VAR FILE

Figures 1 and 2 diagram the internal structure of VAR files.

Variable Storage

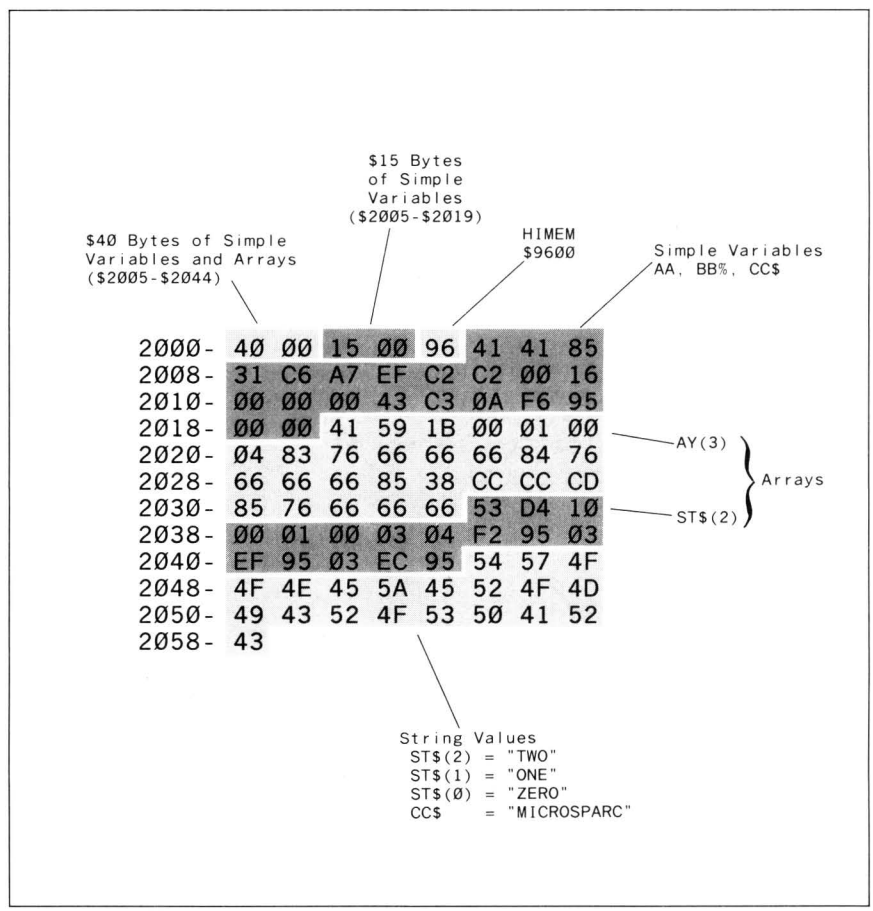
Figure 1 shows how Applesoft variables are stored in memory. Simple variables (as opposed to array variables) are stored immediately above the Applesoft program in memory. Each variable occupies seven bytes, two for the first two characters of the variable name and five for the actual value, in the case of floating point or integer variables, or an address, in the case of string variables or function definitions. (Details on each type of variable can be found in the *Applesoft BASIC Programmer's Reference Manual*.)

FIGURE 1: Applesoft Variable Storage and ProDOS VAR File Structure



Array variables are stored in a separate area in memory, just above the simple variables, and each element of an array has a structure similar to a simple variable of the same type. The actual characters in each

FIGURE 2: VAR File BLOADed at Address $2000



LISTING 1: DISPLAY *(continued)*

```
2016:20 A9 20      85            JSR   SETBIT          ; from ProDOS by setting
2019:E6 3C         86            INC   A1L             ; bits in the bitmap
201B:C6 3D         87            DEC   A1L+1
201D:D0 F5  2014   88            BNE   PROTECT
201F:              89 ;
201F:AD 08 BE      90            LDA   EXTRNCMD+2      ;Link DISPLAY into the
2022:8D 34 22      91            STA   TRYNEXT+2       ; chain of external
2025:AD 07 BE      92            LDA   EXTRNCMD+1      ; commands, if any
2028:8D 33 22      93            STA   TRYNEXT+1
202B:AD 3A 21      94            LDA   ORIG            ;If DISPLAY cannot
202E:8D 08 BE      95            STA   EXTRNCMD+2      ; recognize a command,
2031:A9 00         96            LDA   #0              ; it will pass control
2033:8D 07 BE      97            STA   EXTRNCMD+1      ; to next routine
2036:              98 ;
2036:A9 00         99            LDA   #>BEGIN         ;To start relocation,
2038:85 3A        100            STA   PCL             ; point program
203A:A9 22        101            LDA   #<BEGIN         ; counter at beginning
203C:85 3B        102            STA   PCL+1           ; of program
203E:A2 00        103 FIXLOOP    LDX   #0
2040:20 8C F8     104            JSR   INSDS2          ;Disassemble an opcode
2043:B1 3A        105            LDA   (PCL),Y         ;Check opcode
2045:F0 28  206F  106            BEQ   FIXADDR         ;BRK means end of code
2047:A4 2F        107            LDY   LENGTH          ;Only 3-byte
2049:C0 02        108            CPY   #2              ; instructions need
204B:D0 10  205D  109            BNE   FX1             ; fixing
204D:B1 3A        110            LDA   (PCL),Y         ;Only instructions
204F:C9 22        111            CMP   #<BEGIN         ; referring to address
2051:90 0A  205D  112            BCC   FX1             ; within program
2053:C9 26        113            CMP   #<LAST+$100     ; need fixing
2055:B0 06  205D  114            BCS   FX1
2057:18           115            CLC                   ;Fix by adding RELO
2058:6D 3B 21     116            ADC   RELO            ; offset to hi byte
205B:91 3A        117            STA   (PCL),Y         ; of address
205D:20 53 F9     118 FX1        JSR   PCADJ
2060:85 3A        119            STA   PCL             ;Move program counter
2062:84 3B        120            STY   PCL+1           ; to next instruction
2064:4C 3E 20     121            JMP   FIXLOOP
2067:             122 ;
2067:18           123 FXTBLP     CLC                   ;Add RELO offset to
2068:6D 3B 21     124            ADC   RELO            ; hi byte of each
206B:99 5C 25     125            STA   ATBL,Y          ; address in the table
206E:C8           126            INY                   ;Next address
206F:C8           127 FIXADDR    INY
2070:B9 5C 25     128            LDA   ATBL,Y          ;0 means end of
2073:D0 F2  2067  129            BNE   FXTBLP          ; table
2075:             130 ;
2075:A0 00        131            LDY   #0              ;Move program to
2077:84 3C        132            STY   A1L             ; the space reserved
2079:A9 22        133            LDA   #<BEGIN         ; for it, using the
207B:85 3D        134            STA   A1L+1           ; monitor MOVE routine
207D:A9 CB        135            LDA   #>LAST
207F:85 3E        136            STA   A2L
2081:A9 25        137            LDA   #<LAST
2083:85 3F        138            STA   A2L+1
2085:84 42        139            STY   A4L
2087:AD 3A 21     140            LDA   ORIG
208A:85 43        141            STA   A4L+1
208C:20 2C FE     142            JSR   MOVE
208F:             143 ;
208F:A2 7A        144            LDX   #BANNER1-BANNER0
2091:A0 00        145            LDY   #0
2093:B9 C0 20     146 BNRLP      LDA   BANNER0,Y
2096:2C 98 BF     147            BIT   MACHID          ;Check for Apple II+
2099:30 06  20A1  148            BMI   BNROUT
209B:C9 E0        149            CMP   #$E0            ;Convert lower case
209D:90 02  20A1  150            BCC   BNROUT          ; to upper case
209F:29 DF        151            AND   #%11011111      ; for II+
20A1:20 5C DB     152 BNROUT     JSR   OUTDO           ;Print greeting
20A4:C8           153            INY
20A5:CA           154            DEX
20A6:D0 EB  2093  155            BNE   BNRLP
20A8:60           156            RTS                   ;End of installation
20A9:             157 ;
20A9:48           158 SETBIT     PHA                   ;This subroutine protects
20AA:29 07        159            AND   #$07            ; the memory page whose hi
20AC:AA           160            TAX                   ; address byte is in A
20AD:68           161            PLA
20AE:4A           162            LSR   ;Upper          five bits of A
20AF:4A           163            LSR   ;               make an index into
20B0:4A           164            LSR   ;               the bitmap
20B1:A8           165            TAY
20B2:A9 00        166            LDA   #0              ;Lower three bits of A
20B4:38           167            SEC                   ; choose the bit
20B5:6A           168 SETLP      ROR   ;               to be set--X counts
20B6:CA           169            DEX                   ; while bit is shifted
20B7:10 FC  20B5  170            BPL   SETLP           ; from carry bit
20B9:19 58 BF     171            ORA   BITMAP,Y
20BC:99 58 BF     172            STA   BITMAP,Y        ;Set the bit in the bitmap
20BF:60           173            RTS
20C0:             174 ;
20C0:A0 A0 A0 A0  175 BANNER0    ASC   '                  DISPLAY'
20C4:A0 A0 A0 A0
20C8:A0 A0 A0 A0
20CC:A0 A0 A0 A0
20D0:C4 C9 D3 D0
20D4:CC C1 D9
20D7:8D           176            DB    $8D
```

continued on page 82

string are stored in a separate block of memory just below ProDOS and its buffers.

Figure 1 also shows how these three variable storage areas are put into a variable file. Each area remains intact and separate. The only changes are the deletion of free space and the addition of five bytes at the beginning of the file.

File Information Storage

The five bytes provide three pieces of critical information:

1. The combined size of the simple variable and array areas
2. The size of the simple variable area alone

DISPLAY is compatible with other ProDOS add-on commands that use the recommended installation methods.

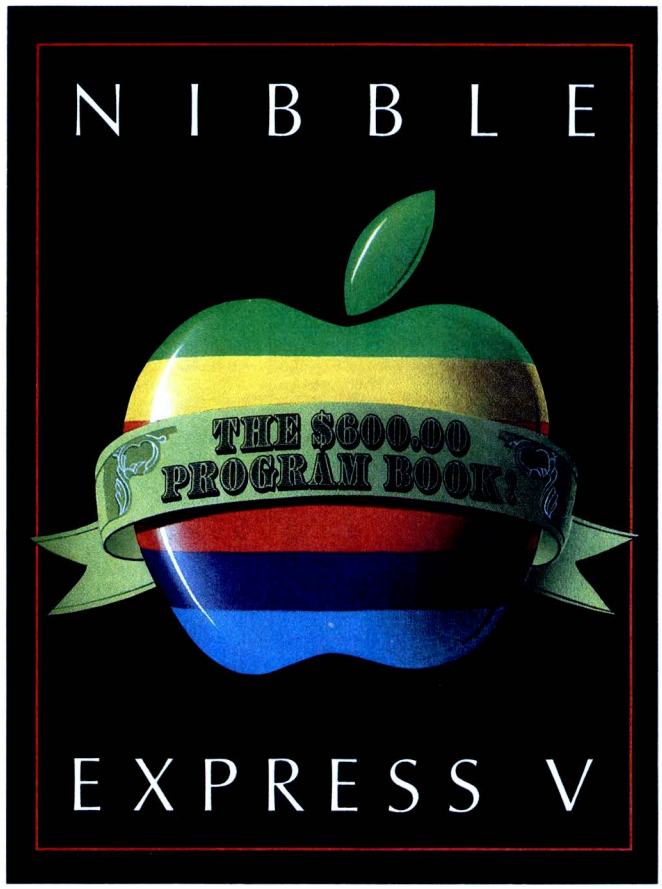
3. The value of the HIMEM page when the VAR file was stored

This information is used by the RESTORE command to reconstruct the variable space.

The usefulness of the third value is not immediately obvious. The value of HIMEM determines where string values are stored; if HIMEM has been changed from the original value when a VAR file is RESTOREd, then the string values will be moved accordingly. But remember that string variables are stored with pointers to a different area of memory where the actual characters are stored. When the string characters are moved, the string pointers must be adjusted to accommodate the new location.

The location of the string pointers can be changed another way. ProDOS stores a load address for the VAR file in the disk catalog. If the file is loaded so that the simple variables start at this address, then all the string pointers are correct. Therefore, the pointers can be corrected by subtracting the difference between the load address stored in the disk directory and the address at which the file is actually loaded. This is, in fact, the method that DISPLAY uses to find the true locations of the string characters.

Figure 2 shows a typical VAR file in detail. The file has three simple variables: AA=22.222, BB%=22, and CC$="MICROSPARC". It also has a real array, AY(3), and a string array, ST$(2). The five bytes at the beginning of the file show that the simple and array variables occupy $40 bytes (from $2005 to $2044), that simple



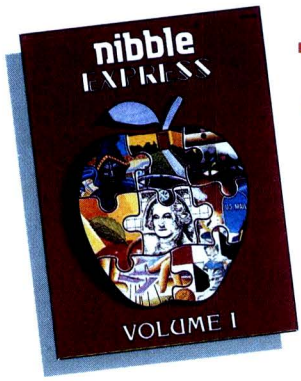
Nibble Express gives you a 3,343% return on your investment!

Nibble Express, an anthology of Nibble Magazine programs and feature articles from an entire year, costs just **$14.95-$19.95.** Yet each volume contains over $600 worth of valuable, time-saving programs. Written by the best authors.

You don't have to be a stock broker to realize this amounts to a 3,343% return on your investment. Here are some highlights:

NIBBLE EXPRESS I

Apple TRAC Plus—Adaptable system for analyzing where your money goes, and examining spending trends. (TRAC Plus is the core program for a complete Personal Finance System with its companion programs TRAC Budget, TRAC Graphics, and Income TRAC, also featured in this ad.)



Apple T.O.U.G.H. (Text Outputter, Updater and Generalized Handler)—Turn your Apple into a line-oriented Text Processor.

Apple Simon—Low-Res Graphics version of the popular game. Test your coordination and memory!

PIP II (Personalized Inventory Program)—provides a complete *disk-based* Data Management System for managing and tracking your personal assets.

Apple A.I.M. (Automated Intelligent Mailing)—Highly modular system for managing Mailing Lists and other Filing/Retrieval/Listing applications.

NIBBLE EXPRESS II

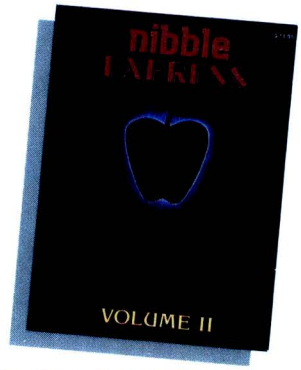
Apple CHAMP (Checkbook Advanced Management Program)—Versatile data entry and Checkbook Reconciliation System

Music Retrieval System (M.R.S)—Fast, flexible file management system for storing, cataloging, and retrieving information on your record collection.

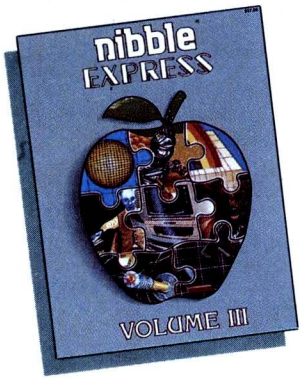
Apple Star System—Self-prompting system for Data File Management.

Apple MAPS—Comprehensive Marketing and Personal Shopping System, helps keep track of grocery stocks.

TRAC "B" (Budget TRAC)—Flexible, fast budget account preparation and reporting system for your personal finances. Works with Apple TRAC.



NIBBLE EXPRESS III *TRAC "I"* (Income TRAC)—Third major module to the TRAC Home Finance System, monitors your income from various sources



Apple MLE (Machine Language Editor)—Powerful, One-of-a-kind utility for entering and editing Machine Language Programs.

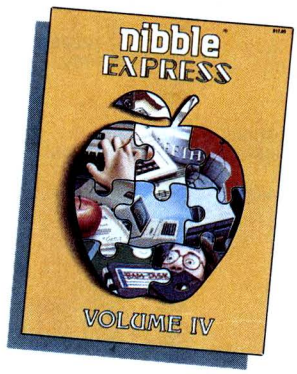
Micro-Calc—Generates your own, customized Finance Program for home or busines.

A.L.E. (Applesoft Line Editor)—Powerful Utility that helps you write and edit Applesoft Programs.

Apple Recipe Box—Makes Cook-Booking simple!

NIBBLE EXPRESS IV

The Nibble Investor—Comprehensive Tracking, Analysis, Reporting, and Graphics Charting System for your Stocks and other investments.



The Nibble RAM Manager—Powerful Utility that puts up to 10 programs of your choice into a 16K RAM Card for instant access at the touch of a key!

T.U.N.E.S.—Comprehensive Music System that lets you create songs and sound effects.

RAM Disk—Makes your RAM card a Pseudo-Disk. Access data files or save and load programs quickly.

NIBBLE EXPRESS V

Nibble File Cabinet—Flexible Database Management System that lets you create variable length records.

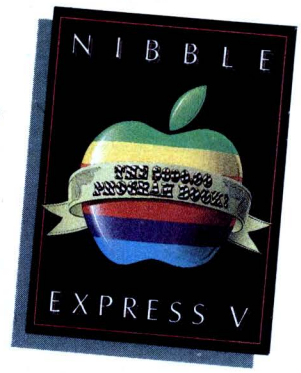
Applesoft Global Editor—Powerful Editor for Applesoft programs offers selected or automatic search and replace.

Sound Synthesizer—Sound Creation Utility that lets you "draw" your sound effects and simple melodies on the Hi-Res screen.

RAM Disk 64—Create a 170 sector pseudodisk on your 128K IIe or IIc for fast access with normal DOS 3.3 commands.

Hi-Res Houdini—Machine Language Graphics Utility creates special graphics effects.

Nibble Diner—Personal Diet Planner provides nutritional analysis of foods and keeps track of your eating habits.



Nibble Garage—Improves car care by tracking maintenance and generating reminders of necessary repairs.

Nibble Broker—Track your stocks and graphically display your profits.

The BASIC Assembler—Editor/Assembler package, written in Applesoft BASIC, produces machine language programs quickly and easily.

Coupman—Make better use of your coupons and save!

After you've made a big investment in your Apple, it pays to keep it busy. Invest in Nibble Express.

Send me NIBBLE EXPRESS!

- ☐ NIBBLE EXPRESS I—$14.95
- ☐ NIBBLE EXPRESS II—$14.95
- ☐ NIBBLE EXPRESS III—$17.95
- ☐ NIBBLE EXPRESS IV—$17.95
- ☐ **NIBBLE EXPRESS V—$19.95**

- ☐ Mastercard
- ☐ Visa
- ☐ Check, M. O.

Please add $1.75 shipping/handling per copy (outside U.S.A. add $2.75 surface or $6.50 Air Mail per copy). Mass. residents add 5% sales tax. Payable in U.S. funds only.

Name _____ Tel. # _____

Address _____

City _____ State _____ Zip _____

Signature _____ Date _____

Charge Card # _____ Exp. Date _____

45 Winthrop St., Concord MA 01742 (617) 371-1660

variables alone occupy $15 bytes (from $2005 to $2019), and that HIMEM was at $9600 when the file was STOREd. Under ProDOS, HIMEM can only be set on even page boundaries. The low byte of HIMEM is always zero, so only the high byte needs to be saved.

Where's the Function?

During my investigations, I made a rather surprising discovery about function definitions in VAR files — they are missing! The function name is stored with the simple variables, but the definition itself is inside the original Applesoft program. If the program that created the definition is not present, or if the program has been changed, the function definition is not available. An attempt to use such a function will evoke a SYNTAX ERROR message. Since DISPLAY will usually be used without a program, it does not attempt to display the function definition. Instead, it prints "Function definition" to indicate that the function name has indeed been defined.

Like other ProDOS commands, DISPLAY can be used from within a program in a PRINT statement preceded by a <CONTROL>D...

HOW DISPLAY WORKS

DISPLAY loads at $2000. The first part of the program is a loader routine that requests space from BASIC.SYSTEM, protects that space from ProDOS, relocates the rest of DISPLAY, and connects it to ProDOS.

The idea for the relocation method came from Tom Weishaar's article, "A Type Command for ProDOS" (*Softalk*, Vol. 4, June, 1984). My implementation differs, however, in that the program code and data must be strictly segregated into separate sections. Furthermore, the code cannot contain instructions that refer to memory locations by immediate addressing (LDA #LOOP or LDA #>LOOP, for instance). Instead, internal addresses are placed in a separate table at the end of the program (**lines 646-651** in **Listing 1**).

The relocation routine has two tasks. First, it adjusts the third byte of every three-byte instruction in the code section, using a Monitor subroutine to identify three-byte instructions. Second, it adjusts the second

LISTING 1: DISPLAY *(continued)*

```
20D8:A0 A0 A0 C4   177            ASC   '   Displays the values in'
20DC:E9 F3 F0 EC
20E0:E1 F9 F3 A0
20E4:F4 E8 E5 A0
20E8:F6 E1 EC F5
20EC:E5 F3 A0 E9
20F0:EE
20F1:A0 E1 A0 D6   178            ASC   ' a VAR file'
20F5:C1 D2 A0 E6
20F9:E9 EC E5
20FC:8D            179            DB    $8D
20FD:A0 A0 A0 A0   180            ASC   '              by Ken Manly'
2101:A0 A0 A0 A0
2105:A0 A0 A0 A0
2109:A0 E2 F9 A0
210D:CB E5 EE A0
2111:CD E1 EE EC
2115:F9
2116:8D            181            DB    $8D
2117:A0 A0 A0 A0   182            ASC   '       (C) 1986 by '
211B:A0 A0 A8 C3
211F:A9 A0 B1 B9
2123:B8 B6 A0 E2
2127:F9 A0
2129:CD E9 E3 F2   183            ASC   'MicroSPARC,  Inc.'
212D:EF D3 D0 C1
2131:D2 C3 AC A0
2135:C9 EE E3 AE
2139:8D            184            DB    $8D
213A:         213A 185 BANNER1    EQU   *
213A:         0001 186 ORIG       DS    1
213B:         0001 187 RELO       DS    1
213C:              188 ;
213C:         003C 189 HERE       EQU   >*
213C:         00C4 190            DS    $100-HERE
2200:              191 ;
2200:A2 00         192 BEGIN      LDX   #0              ;Compare command in the
2202:BD 00 02      193 NXTCHR     LDA   INPUT,X         ; input buffer with
2205:DD 86 25      194            CMP   CMD+1,X         ; our command (DISPLAY)
2208:D0 27   2231  195            BNE   NOTOURS         ;Mismatch--not our cmd
220A:E8            196            INX
220B:EC 85 25      197            CPX   CMD
220E:D0 F2   2202  198            BNE   NXTCHR
2210:              199 ;
2210:CA            200            DEX                   ;Command matched
2211:8E 52 BE      201            STX   XLEN            ;The length of our cmd
2214:AD 5C 25      202            LDA   T.BEGIN0        ;Tell ProDOS where
2217:8D 50 BE      203            STA   XTRNADDR        ; the execution code
221A:AD 5D 25      204            LDA   T.BEGIN0+1      ; for this command is
221D:8D 51 BE      205            STA   XTRNADDR+1
2220:A9 00         206            LDA   #0              ;Indicate external command
2222:8D 53 BE      207            STA   XCNUM
2225:A9 81         208            LDA   #%10000001      ;Tell parser to use prefix
2227:8D 54 BE      209            STA   PBITS           ; and expect a filename
222A:A9 04         210            LDA   #%00000100      ;Tell parser to allow
222C:8D 55 BE      211            STA   PBITS+1         ; slot & drive number
222F:18            212            CLC                   ;Tell ProDOS command is OK
2230:60            213            RTS
2231:              214 ;
2231:38            215 NOTOURS    SEC                   ;Tell ProDOS command is
2232:4C 9E BE      216 TRYNEXT    JMP   XRETURN         ; not recognized
2235:              217 ;
2235:20 C6 22      218 BEGIN0     JSR   CHECKFILE       ;Look for VAR file
2238:B0 18   2252  219            BCS   ERROR
223A:20 E0 22      220            JSR   OPENFILE        ;Get buffer & open file
223D:B0 13   2252  221            BCS   ERROR
223F:20 F0 22      222            JSR   READFILE        ;Read entire file
2242:B0 0E   2252  223            BCS   ERROR
2244:20 19 23      224            JSR   CLOSEFILE
2247:20 25 23      225            JSR   SETUP           ;Set pointers
224A:20 6D 23      226            JSR   PRINTSIMP       ;Print simple variables
224D:20 84 24      227            JSR   PRTARRAY        ;Print arrays
2250:18            228            CLC                   ;Everything OK
2251:60            229            RTS                   ;That's all, folks . .
2252:              230 ;
2252:48            231 ERROR      PHA                   ;In case of error
2253:20 FB DA      232            JSR   CRDO
2256:A2 18         233            LDX   #MSG1-MSG0      ;Print message
2258:A0 00         234            LDY   #0
225A:20 7C 22      235            JSR   MESSAGE
225D:AD 6C BE      236            LDA   VPATH1          ;Reprint requested
2260:85 FE         237            STA   PRTPTR          ; pathname
2262:AD 6D BE      238            LDA   VPATH1+1
2265:85 FF         239            STA   PRTPTR+1
2267:A0 00         240            LDY   #0
2269:B1 FE         241            LDA   (PRTPTR),Y
226B:AA            242            TAX
226C:C8            243            INY
226D:20 92 22      244            JSR   PRTLOOP
2270:20 FB DA      245            JSR   CRDO
2273:20 B0 22      246            JSR   BELL            ;Make a ProDOS bell
2276:68            247            PLA                   ;Recover error code
2277:20 0C BE      248            JSR   PRINTERR        ;Let ProDOS print its
227A:18            249            CLC                   ; error message
227B:60            250            RTS
227C:              251 ;
227C:         227C 252 MESSAGE    EQU   *               ;General purpose
```

```
227C:B9 8D 25      253            LDA   MSG0,Y        ; message printer
227F:2C 98 BF      254            BIT   MACHID        ;Check for Apple II+
2282:30 06   228A  255            BMI   CHROUT
2284:C9 E0         256            CMP   #$E0          ;Convert lower case
2286:90 02   228A  257            BCC   CHROUT        ; to upper case
2288:29 DF         258            AND   #%11011111    ; for II+
228A:20 5C DB      259 CHROUT     JSR   OUTDO
228D:C8            260            INY                 ;Uses Y to find char
228E:CA            261            DEX                 ; and X to count length
228F:D0 EB   227C  262            BNE   MESSAGE
2291:60            263            RTS
2292:              264 ;
2292:B1 FE         265 PRTLOOP    LDA   (PRTPTR),Y    ;Printer for strings
2294:09 80         266            ORA   #$80          ; or pathname
2296:C9 A0         267            CMP   #$A0
2298:B0 0E   22A8  268            BCS   PRT
229A:48            269            PHA
229B:A9 DB         270            LDA   #'[           ;Prints control chars
229D:20 5C DB      271            JSR   OUTDO         ; inside square brackets
22A0:68            272            PLA
22A1:09 40         273            ORA   #$40
22A3:20 5C DB      274            JSR   OUTDO
22A6:A9 DD         275            LDA   #']
22A8:20 5C DB      276 PRT        JSR   OUTDO
22AB:C8            277            INY
22AC:CA            278            DEX
22AD:D0 E3   2292  279            BNE   PRTLOOP
22AF:60            280            RTS
22B0:              281 ;
22B0:        22B0  282 BELL       EQU   *
22B0:A2 20         283            LDX   #$20
22B2:A9 02         284 BELL1      LDA   #$2           ;Pretty ProDOS bell
22B4:20 A8 FC      285            JSR   WAIT
22B7:8D 30 C0      286            STA   CLICK         ;Suggested by ProDOS
22BA:A9 24         287            LDA   #$24          ; Tech Ref Manual
22BC:20 A8 FC      288            JSR   WAIT
22BF:8D 30 C0      289            STA   CLICK
22C2:CA            290            DEX
22C3:D0 ED   22B2  291            BNE   BELL1
22C5:60            292            RTS
22C6:              293 ;
22C6:        22C6  294 CHECKFILE  EQU   *
22C6:A9 0A         295            LDA   #$0A          ;Ten parameters in
22C8:8D B4 BE      296            STA   SSGINFO       ; parameter list
22CB:A9 C4         297            LDA   #GETINFO      ;Call MLI through
22CD:20 70 BE      298            JSR   GOSYSTEM      ; BASIC.SYSTEM global page
22D0:B0 0D   22DF  299            BCS   CHKOUT
22D2:AD B8 BE      300            LDA   FIFILID       ;Check file type
22D5:C9 FD         301            CMP   #$FD          ;VAR file
22D7:D0 03   22DC  302            BNE   MISMTCH
22D9:18            303            CLC
22DA:90 03   22DF  304            BCC   CHKOUT
22DC:A9 0D         305 MISMTCH    LDA   #13           ;File type mismatch
22DE:38            306            SEC
22DF:60            307 CHKOUT     RTS
22E0:              308 ;
22E0:A5 74         309 OPENFILE   LDA   HIMEM+1       ;Use ProDOS temp buffer
22E2:8D CF BE      310            STA   OSYSBUF+1     ; which is just above
22E5:A9 00         311            LDA   #0            ; HIMEM
22E7:8D CE BE      312            STA   OSYSBUF
22EA:A9 C8         313            LDA   #OPEN
22EC:20 70 BE      314            JSR   GOSYSTEM
22EF:60            315 OPENRTN    RTS
22F0:              316 ;
22F0:        22F0  317 READFILE   EQU   *
22F0:A6 6A         318            LDX   VARTAB+1      ;Locate data buffer
22F2:E8            319            INX                 ; just above the
22F3:8E D8 BE      320            STX   RWDATA+1      ; BASIC program, if
22F6:8E 75 25      321            STX   BUFFER+1      ; there is one, and
22F9:A2 00         322            LDX   #0            ; starting on a
22FB:8E D7 BE      323            STX   RWDATA        ; page boundary
22FE:8E 74 25      324            STX   BUFFER
2301:AD BC BE      325            LDA   FIBLOKS       ;Get file length in blocks
2304:0A            326            ASL   ;             and tell ProDOS to read
2305:8D DA BE      327            STA   RWCOUNT+1     ; twice as many pages
2308:A9 00         328            LDA   #0
230A:8D D9 BE      329            STA   RWCOUNT
230D:AD D0 BE      330            LDA   OREFNUM       ;Get file reference number
2310:8D D6 BE      331            STA   RWREFNUM      ; from OPEN parm list
2313:A9 CA         332            LDA   #READ
2315:20 70 BE      333            JSR   GOSYSTEM
2318:60            334            RTS
2319:              335 ;
2319:        2319  336 CLOSEFILE  EQU   *
2319:AD D0 BE      337            LDA   OREFNUM       ;File reference number
231C:8D DE BE      338            STA   CREFNUM       ; from OPEN parm list
231F:A9 CC         339            LDA   #CLOSE
2321:20 70 BE      340            JSR   GOSYSTEM
2324:60            341            RTS
2325:              342 ;
2325:        2325  343 SETUP      EQU   *
2325:38            344            SEC                 ;Difference between original
2326:AD B9 BE      345            LDA   FIAUXID       ; file load point (FIAUXID)
2329:E9 05         346            SBC   #5            ; and present location
232B:8D 7A 25      347            STA   OFFSET        ; (BUFFER+5) is OFFSET,
232E:AD BA BE      348            LDA   FIAUXID+1     ; which will be used to
2331:ED 75 25      349            SBC   BUFFER+1      ; correct string pointers
2334:8D 7B 25      350            STA   OFFSET+1
2337:              351 ;
```

continued on next page

byte of every address in the internal address table. In either case, the relocation routine adjusts the appropriate byte by adding to it the difference between the program's present location and its final location (RELO).

Once DISPLAY is installed, its operation is relatively simple. If ProDOS receives a command it does not recognize, it transfers program control to DISPLAY. If the command is DISPLAY, then DISPLAY returns control to ProDOS, which interprets the rest of the command (the file name, slot, and drive). ProDOS transfers control back to DISPLAY (at BEGIN0 in **line 218** of **Listing 1**). DISPLAY loads the entire VAR file into memory beginning at a point just above any BASIC program. It will not disturb the program itself, although it may alter its variables. A pointer is set to the beginning of the file and an Applesoft routine is used to move through the file, one variable at a time, printing its name and value. Each time DISPLAY comes to a string variable, it subtracts the OFFSET value from the string pointer to find the address where string characters actually start.

I hope you will find DISPLAY to be a valuable tool in programming the powerful VAR files into your ProDOS programs. It should help both in constructing VAR files and in debugging ProDOS Applesoft programs.

LISTING 1: DISPLAY *(continued)*

```
2337:AD 74 25       352             LDA   BUFFER          ;Put buffer location
233A:85 3C          353             STA   A1L             ; on zero page
233C:AD 75 25       354             LDA   BUFFER+1        ; for indirect addressing
233F:85 3D          355             STA   A1L+1
2341:               356 ;
2341:18             357             CLC                   ;First two bytes in buffer
2342:A9 05          358             LDA   #5              ; have the total no of bytes
2344:A0 00          359             LDY   #0              ; of simple and array vars--
2346:71 3C          360             ADC   (A1L),Y         ;Add this number to start of
2348:8D 78 25       361             STA   ARRYEND         ; vars (BUFFER+5) to get end
234B:AD 75 25       362             LDA   BUFFER+1        ; of array variables
234E:C8             363             INY
234F:71 3C          364             ADC   (A1L),Y
2351:8D 79 25       365             STA   ARRYEND+1
2354:               366 ;
2354:18             367             CLC
2355:A9 05          368             LDA   #5              ;Third and fourth bytes
2357:C8             369             INY                   ; in buffer have total
2358:71 3C          370             ADC   (A1L),Y         ; no of bytes of simple
235A:8D 76 25       371             STA   SIMPEND         ; vars--add this to
235D:8D 7D 25       372             STA   NXTARRY         ; the beginning of vars
2360:AD 75 25       373             LDA   BUFFER+1        ; to get the end of
2363:C8             374             INY                   ; simple vars and the
2364:71 3C          375             ADC   (A1L),Y         ; location of the first
2366:8D 77 25       376             STA   SIMPEND+1       ; array variable
2369:8D 7E 25       377             STA   NXTARRY+1
236C:60             378             RTS
236D:               379 ;
236D:       236D    380 PRINTSIMP   EQU   *
236D:A9 05          381             LDA   #5              ;Start printing simple
236F:85 FC          382             STA   BUFPTR          ; variables at BUFFER+5
2371:AD 75 25       383             LDA   BUFFER+1
2374:85 FD          384             STA   BUFPTR+1
2376:4C D6 23       385             JMP   CHKSIMP
2379:A0 00          386 SIMPLOOP    LDY   #0              ;Clear variable type byte
237B:8C 7C 25       387             STY   VARTYPE
237E:B1 FC          388             LDA   (BUFPTR),Y      ;First char of variable
2380:0A             389             ASL   ;Save           high bit
2381:2E 7C 25       390             ROL   VARTYPE         ; in type byte
2384:38             391             SEC
2385:6A             392             ROR   ;Set            high bit
2386:20 5C DB       393             JSR   OUTDO           ;Print char
2389:20 A2 24       394             JSR   INCBP           ;Point at next char
238C:B1 FC          395             LDA   (BUFPTR),Y      ;Second char of variable
238E:20 A2 24       396             JSR   INCBP           ;Point at next byte
2391:0A             397             ASL   ;Save           high bit of char
2392:2E 7C 25       398             ROL   VARTYPE         ; in type byte
2395:38             399             SEC                   ;Set high bit
2396:6A             400             ROR
2397:C9 A0          401             CMP   #$A0            ;Check for null or control
2399:B0 02   239D   402             BCS   OK
239B:A9 A0          403             LDA   #$A0            ;If so, replace with space
239D:20 5C DB       404 OK          JSR   OUTDO           ;Print it
23A0:AC 7C 25       405             LDY   VARTYPE         ;Use type byte as an index
23A3:B9 81 25       406             LDA   VARSYMB,Y       ; to print the symbol
23A6:20 5C DB       407             JSR   OUTDO           ; corresponding to the type
23A9:A2 03          408             LDX   #MSG2-MSG1
23AB:A0 18          409             LDY   #MSG1-MSG0
23AD:20 7C 22       410             JSR   MESSAGE         ;Print ' = '
23B0:               411 ;
23B0:AD 5F 25       412             LDA   T.SIMPCONT+1    ;Load the stack with
23B3:48             413             PHA                   ; the SIMPCONT address
23B4:AD 5E 25       414             LDA   T.SIMPCONT      ; so that control will
23B7:48             415             PHA                   ; go there after funny JSR
23B8:AD 7C 25       416             LDA   VARTYPE         ;Use type byte as an
23BB:0A             417             ASL   ;               index to get the address
23BC:A8             418             TAY                   ; of the routine which
23BD:B9 63 25       419             LDA   SIMPTBL+1,Y     ; will print the variable--
23C0:48             420             PHA                   ;Put the routine's address
23C1:B9 62 25       421             LDA   SIMPTBL,Y       ; on the stack and call the
23C4:48             422             PHA                   ; routine by doing an RTS
23C5:60             423             RTS                   ;Funny JSR
23C6:20 FB DA       424 SIMPCONT    JSR   CRDO
23C9:18             425             CLC
23CA:A5 FC          426             LDA   BUFPTR          ;Advance pointer
23CC:69 05          427             ADC   #5              ; to the next variable
23CE:85 FC          428             STA   BUFPTR
23D0:A5 FD          429             LDA   BUFPTR+1
23D2:69 00          430             ADC   #0
23D4:85 FD          431             STA   BUFPTR+1
23D6:               432 ;
23D6:A5 FD          433 CHKSIMP     LDA   BUFPTR+1        ;Has the buffer pointer
23D8:CD 77 25       434             CMP   SIMPEND+1       ; reached the end of the
23DB:90 9C   2379   435             BCC   SIMPLOOP        ; simple variables?
23DD:A5 FC          436             LDA   BUFPTR
23DF:CD 76 25       437             CMP   SIMPEND
23E2:90 95   2379   438             BCC   SIMPLOOP        ; If not, print another
23E4:60             439             RTS                   ; If so, return
23E5:               440 ;
23E5:       23E5    441 ARRAYLOOP   EQU   *
23E5:A0 00          442             LDY   #0              ;Clear variable type byte
23E7:8C 7C 25       443             STY   VARTYPE
23EA:B1 FC          444             LDA   (BUFPTR),Y      ;First char of variable
23EC:0A             445             ASL   ;Save           high bit
23ED:2E 7C 25       446             ROL   VARTYPE         ; in type byte
23F0:38             447             SEC
23F1:6A             448             ROR   ;Set            high bit
```

```
23F2:20 5C DB      449            JSR   OUTDO          ;Print char
23F5:C8            450            INY                  ;Point at next char
23F6:B1 FC         451            LDA   (BUFPTR),Y     ;Second char of variable
23F8:0A            452            ASL   ;Save          high bit of char
23F9:2E 7C 25      453            ROL   VARTYPE        ; in type byte
23FC:38            454            SEC                  ;Set high bit
23FD:6A            455            ROR
23FE:20 5C DB      456            JSR   OUTDO          ;Print char
2401:AC 7C 25      457            LDY   VARTYPE        ;Use type bit as an index
2404:F0 06   240C  458            BEQ   PAREN          ;No symbol for real array
2406:B9 81 25      459            LDA   VARSYMB,Y      ;Get and print symbol
2409:20 5C DB      460            JSR   OUTDO          ; corresponding to type
240C:A9 A8         461 PAREN      LDA   #'(
240E:20 5C DB      462            JSR   OUTDO
2411:              463 ;
2411:A0 02         464            LDY   #2             ;Update the pointer to
2413:18            465            CLC                  ; the next array variable
2414:B1 FC         466            LDA   (BUFPTR),Y     ; by adding the offset
2416:65 FC         467            ADC   BUFPTR         ; stored with this variable
2418:8D 7D 25      468            STA   NXTARRY
241B:C8            469            INY
241C:B1 FC         470            LDA   (BUFPTR),Y
241E:65 FD         471            ADC   BUFPTR+1
2420:8D 7E 25      472            STA   NXTARRY+1
2423:C8            473            INY
2424:B1 FC         474            LDA   (BUFPTR),Y     ;Get number of dimensions
2426:8D 7F 25      475            STA   DIM            ; for this variable
2429:              476 ;
2429:18            477            CLC
242A:A5 FC         478            LDA   BUFPTR         ;Move the pointer to
242C:69 04         479            ADC   #4             ; the point at which
242E:85 FC         480            STA   BUFPTR         ; the dimension sizes
2430:A5 FD         481            LDA   BUFPTR+1       ; start
2432:69 00         482            ADC   #0
2434:85 FD         483            STA   BUFPTR+1
2436:              484 ;
2436:AD 7F 25      485            LDA   DIM            ;There are two bytes
2439:0A            486            ASL   ;              for each dimension
243A:8D 80 25      487            STA   DIMPTR         ; to hold the size
243D:D0 05   2444  488            BNE   PRTDIM         ;Branch always
243F:A9 AC         489 DIMLOOP    LDA   #',            ;Print a comma
2441:20 5C DB      490            JSR   OUTDO          ; between dimensions
2444:AC 80 25      491 PRTDIM     LDY   DIMPTR         ;Use dimension pointer
2447:20 F7 24      492            JSR   DIMEN          ; as index to print size
244A:CE 80 25      493            DEC   DIMPTR
244D:CE 80 25      494            DEC   DIMPTR         ;Next dimension
2450:D0 ED   243F  495            BNE   DIMLOOP        ; if there are any left
2452:A2 04         496            LDX   #MSG3-MSG2     ;Print ') = '
2454:A0 1B         497            LDY   #MSG2-MSG0
2456:20 7C 22      498            JSR   MESSAGE
2459:20 FB DA      499            JSR   CRDO           ;And carriage return
245C:              500 ;
245C:AD 7F 25      501            LDA   DIM            ;Move the pointer
245F:0A            502            ASL   ;              to where the data
2460:38            503            SEC                  ; starts
2461:65 FC         504            ADC   BUFPTR         ; (1 + 2 * no of dimen)
2463:85 FC         505            STA   BUFPTR
2465:A9 00         506            LDA   #0
2467:65 FD         507            ADC   BUFPTR+1
2469:85 FD         508            STA   BUFPTR+1
246B:              509 ;
246B:AD 61 25      510            LDA   T.ARRYCONT+1   ;Load the stack with
246E:48            511            PHA                  ; the ARRYCONT address
246F:AD 60 25      512            LDA   T.ARRYCONT     ; so that control will
2472:48            513            PHA                  ; go there after funny JSR
2473:AD 7C 25      514            LDA   VARTYPE        ;Use type byte as an
2476:0A            515            ASL   ;              index to get the address
2477:A8            516            TAY                  ; of the routine which
2478:B9 6B 25      517            LDA   ARRYTBL+1,Y    ; will print the variable--
247B:48            518            PHA                  ;Put the routine's address
247C:B9 6A 25      519            LDA   ARRYTBL,Y      ; on the stack and call the
247F:48            520            PHA                  ; routine by doing an RTS
2480:60            521            RTS                  ;Funny JSR again
2481:20 FB DA      522 ARRYCONT   JSR   CRDO
2484:AD 7E 25      523 PRTARRAY   LDA   NXTARRY+1       ;Has the buffer pointer
2487:CD 79 25      524            CMP   ARRYEND+1      ; reached the end of the
248A:90 09   2495  525            BCC   ARRYLOOP       ; array variables?
248C:AD 7D 25      526            LDA   NXTARRY
248F:CD 78 25      527            CMP   ARRYEND
2492:90 01   2495  528            BCC   ARRYLOOP       ;If not, print another
2494:60            529            RTS                  ;If so, return
2495:              530 ;
2495:AD 7D 25      531 ARRYLOOP   LDA   NXTARRY        ;Set up buffer pointer
2498:85 FC         532            STA   BUFPTR         ; to point to the next
249A:AD 7E 25      533            LDA   NXTARRY+1      ; array
249D:85 FD         534            STA   BUFPTR+1
249F:4C E5 23      535            JMP   ARRAYLOOP
24A2:              536 ;
24A2:E6 FC         537 INCBP      INC   BUFPTR         ;Increment pointer
24A4:D0 02   24A8  538            BNE   INCBP1
24A6:E6 FD         539            INC   BUFPTR+1
24A8:60            540 INCBP1     RTS
24A9:              541 ;
24A9:        24A9  542 REAL       EQU   *
24A9:A5 FC         543            LDA   BUFPTR         ;Transfer buffer pointer
24AB:85 5E         544            STA   INDEX          ; to pointer used by
24AD:A5 FD         545            LDA   BUFPTR+1       ; move routine
24AF:85 5F         546            STA   INDEX+1
```

continued on next page

Apple Pascal 1.2/128K Library

Double Hi-Res graphics (pages 1/1X & 2/2X) + Lo-Res + Double Lo-Res + Hi-Res (pages 1 & 2); all that is possible when using Apple Pascal 1.2,™ version 128K, upgraded by "Apple Pascal 1.2/128K Library". Even better, you'll be able to use 132K of the Apple IIe 128K.

Five intrinsic units to link into SYSTEM.LIBRARY and one regular unit dealing with Double Hi-Res graphics. Two double-sided and one single-sided disks hold text & code of five intrinsic units and a regular one; plus text version of some assembly-language routines used by those units; plus demo program(s) for each unit; plus several FOTO files, etc.

Also on disk, a program to create your own graphics character set. You can load a set from disk, save a set on disk, print a complete character set, print a set side by side with an enlarged character you have chosen, etc.

If you plan to display Double Hi-Res graphics using a Hi-Res color monitor, to take advantage of the 16 colors provided by this mode, you'll find out much useful information in a 7-page tutorial. Seven sheets of graph paper give the equivalence between all the addresses of the Hi-Res page 1 vs the Y-coordinate & the byte/column #.

A 32-page booklet describes all the procedures & functions of the five intrinsic units, explains the purpose of the demo programs and gives many tips and useful addresses (a 3-page appendix).

PEEKLIB unit: PEEK, POKE, doublepeek and doublepoke with addresses in dec or hex, at your choice; two procs to change bits within a byte anywhere in main memory; proc to write in flashing mode with the 80-column card ON (ASCII 64 to 95), etc.

ECRITO unit: 11 procs to use easily the text capabilities of the old ADM, the Imagewriter, or even the Scribe thermal-transfer printer.

RECO unit: to read a disk directory or read/write a file at running time.

LORES unit: to plot blocks in 16 colors on the Lo-Res graphics page (full/mixed screen), save on disk a Lo-Res picture, print one or two side by side Lo-Res image(s).

TRIO unit: 15 graphics procs. PALETTE (18 options) is a menu-driven proc allowing on both normal HGR pages to: load, see or save an image; print page(s) in several modes (even clipping & merged pages). Option to type on any page, passing over pixels without damaging them. You can make a clip on any page and transfer that on the other page. You can also on both pages: scroll a picture in four directions, merge pages in two modes, inverse colors, save any HGR page in aux RAM and re-load that one on any page in main RAM, etc.

Regular unit DUO:
22 procedures dealing with Double Hi-Res graphics. Allows you on page 1/1X or 2/2X to: load from disk a Double Hi-Res picture (32 blocks), save on disk a picture, clear any page, print a picture, inverse colors, merge pages in two modes, scroll a Double Hi-Res picture to the right or the left. You can also transfer any page from main to aux RAM, or the opposite (you can even transfer only a single byte). Proc to merge pages 1/1X & 2/2X (EOR or ORA) and save the result in aux memory, keeping both pages intact; another proc brings back that saved image on page 1/1X or 2/2X.

Send $95.00 in US currency (no credit card) to:

Gilles Asselin,
5734 de Laroche,
Montreal, Que.,
Canada H2S 2C6.

Apple Pascal 1.2, Apple IIe, Imagewriter and Scribe are trademarks of Apple Computer, Inc.

LISTING 1: DISPLAY *(continued)*

```
24B1:20 FD EA      547            JSR   MOVFM+4        ;Move number to FAC
24B4:20 2E ED      548            JSR   PRNTFAC        ;Print value of FAC
24B7:60            549            RTS
24B8:        24B8  550 STRING     EQU   *
24B8:A0 00         551            LDY   #0             ;Get length
24BA:B1 FC         552            LDA   (BUFPTR),Y     ; of string
24BC:AA            553            TAX                  ; and keep in X
24BD:F0 17   24D6  554            BEQ   NULL           ;Can't print empty strings
24BF:C8            555            INY
24C0:38            556            SEC
24C1:B1 FC         557            LDA   (BUFPTR),Y     ;Correct the string pointer
24C3:ED 7A 25      558            SBC   OFFSET         ; by subtracting OFFSET,
24C6:85 FE         559            STA   PRTPTR         ; then store it in PRTPTR
24C8:C8            560            INY                  ; where the printing
24C9:B1 FC         561            LDA   (BUFPTR),Y     ; subroutine can use it
24CB:ED 7B 25      562            SBC   OFFSET+1
24CE:85 FF         563            STA   PRTPTR+1
24D0:A0 00         564            LDY   #0
24D2:20 92 22      565            JSR   PRTLOOP        ;Print the string
24D5:60            566            RTS
24D6:A2 0C         567 NULL       LDX   #MSG5-MSG4     ;Print message
24D8:A0 32         568            LDY   #MSG4-MSG0     ; for empty string
24DA:20 7C 22      569            JSR   MESSAGE
24DD:60            570            RTS
24DE:        24DE  571 INTEGER    EQU   *
24DE:A0 00         572            LDY   #0
24E0:B1 FC         573            LDA   (BUFPTR),Y     ;Load integer value
24E2:AA            574            TAX                  ; into Y and A.
24E3:C8            575            INY
24E4:B1 FC         576            LDA   (BUFPTR),Y
24E6:A8            577            TAY
24E7:8A            578            TXA
24E8:20 F3 DE      579            JSR   INTOFAC1       ;Convert to FP
24EB:20 2E ED      580            JSR   PRNTFAC        ;Print FAC
24EE:60            581            RTS
24EF:        24EF  582 FUNCTION   EQU   *
24EF:A2 13         583            LDX   #MSG4-MSG3     ;Function definition is
24F1:A0 1F         584            LDY   #MSG3-MSG0     ; not stored in a VAR file--
24F3:20 7C 22      585            JSR   MESSAGE        ;Print a message instead of
24F6:60            586            RTS                  ; the definition
24F7:              587 ;
24F7:        24F7  588 DIMEN      EQU   *
24F7:B1 FC         589            LDA   (BUFPTR),Y     ;Load dimension
24F9:AA            590            TAX                  ; size
24FA:88            591            DEY
24FB:B1 FC         592            LDA   (BUFPTR),Y
24FD:CA            593            DEX                  ;Dimension size must
24FE:E0 FF         594            CPX   #$FF           ; be decremented before
2500:D0 03   2505  595            BNE   DIMEN1         ; printing
2502:38            596            SEC
2503:E9 01         597            SBC   #1
2505:20 24 ED      598 DIMEN1     JSR   LINPRT
2508:60            599            RTS
2509:              600 ;
2509:20 A9 24      601 REALARRY   JSR   REAL           ;Print a real number
250C:20 FB DA      602            JSR   CRDO           ; and a carriage return
250F:A9 05         603            LDA   #5             ;Advance 5 bytes to get
2511:20 33 25      604            JSR   NXTELMNT       ; the next number
2514:90 F3   2509  605            BCC   REALARRY       ;Do it again if any are left
2516:60            606            RTS
2517:              607 ;
2517:20 B8 24      608 STRARRY    JSR   STRING         ;Print a string
251A:20 FB DA      609            JSR   CRDO           ; and a carriage return
251D:A9 03         610            LDA   #3             ;Advance 3 bytes to get
251F:20 33 25      611            JSR   NXTELMNT       ; the next string
2522:90 F3   2517  612            BCC   STRARRY        ;Do it again if any are left
2524:60            613            RTS
2525:              614 ;
2525:20 DE 24      615 INTARRY    JSR   INTEGER        ;Print an integer
2528:20 FB DA      616            JSR   CRDO           ; and a carriage return
252B:A9 02         617            LDA   #2             ;Advance 2 bytes to get
252D:20 33 25      618            JSR   NXTELMNT       ; the next number
2530:90 F3   2525  619            BCC   INTARRY        ;Do it again if any are left
2532:60            620            RTS
2533:              621 ;
2533:18            622 NXTELMNT   CLC
2534:65 FC         623            ADC   BUFPTR         ;Add the contents of A
2536:85 FC         624            STA   BUFPTR         ; to the buffer pointer
2538:A9 00         625            LDA   #0
253A:65 FD         626            ADC   BUFPTR+1
253C:85 FD         627            STA   BUFPTR+1
253E:              628 ;
253E:AD 00 C0      629            LDA   KBD            ;Check the keyboard for keys
2541:10 0B   254E  630            BPL   NXTLMNT0       ; that abort array printing
2543:8D 10 C0      631            STA   KBSTROBE
2546:C9 83         632            CMP   #$83           ;Cntrl-C
2548:F0 10   255A  633            BEQ   NXTLMNT1
254A:C9 9B         634            CMP   #$9B           ;Escape
254C:F0 0C   255A  635            BEQ   NXTLMNT1
254E:              636 ;
254E:A5 FD         637 NXTLMNT0   LDA   BUFPTR+1
2550:CD 7E 25      638            CMP   NXTARRY+1      ;Has buffer pointer reached
2553:90 05   255A  639            BCC   NXTLMNT1       ; the next array?
2555:A5 FC         640            LDA   BUFPTR
2557:CD 7D 25      641            CMP   NXTARRY
255A:60            642 NXTLMNT1   RTS                  ;Carry set if so
255B:              643 ;
255B:00            644            DB    0              ;Tell relocater table follows
```

```
255C:                645 ;
255C:          255C  646 ATBL      EQU   *              ;Internal addresses
255C:35 22           647 T.BEGINØ  DW    BEGINØ
255E:C5 23           648 T.SIMPCONT DW   SIMPCONT-1
2560:80 24           649 T.ARRYCONT DW   ARRYCONT-1
2562:A8 24           650 SIMPTBL   DW    REAL-1,STRING-1,FUNCTION-1,INTEGER-1
2564:B7 24
2566:EE 24
2568:DD 24
256A:08 25           651 ARRYTBL   DW    REALARRY-1,STRARRY-1,BELL-1,INTARRY-1
256C:16 25
256E:AF 22
2570:24 25
2572:00 00           652           DW    Ø              ;Tells the relocater to quit
2574:                653 ;
2574:          0002  654 BUFFER    DS    2              ;Start of main data buffer
2576:          0002  655 SIMPEND   DS    2              ;End of simple vars--
2578:                656 ;                              start of arrays
2578:          0002  657 ARRYEND   DS    2              ;End of arrays--
257A:                658 ;                              start of string values
257A:          0002  659 OFFSET    DS    2              ;Correction for
257C:                660 ;                              string pointers
257C:          0001  661 VARTYPE   DS    1              ;Type of variable (Ø-3)
257D:                662 ;                              real,string,funct,int
257D:          0002  663 NXTARRY   DS    2              ;Beginning of array
257F:                664 ;                              after the current one
257F:          0001  665 DIM       DS    1              ;Number of dimensions
2580:                666 ;                              in array
2580:          0001  667 DIMPTR    DS    1              ;Pointer to print size
2581:                668 ;                              of each dimension
2581:A0 A4 A0 A5     669 VARSYMB   ASC   .              $ %. ;Symbols for
2585:                670 ;                              the four variable types
2585:07 C4 C9 D3     671 CMD       STR   'DISPLAY'
2589:D0 CC C1 D9
258D:D5 EE E1 E2     672 MSGØ      ASC   'Unable to read the file '
2591:EC E5 A0 F4
2595:EF A0 F2 E5
2599:E1 E4 A0 F4
259D:E8 E5 A0 E6
25A1:E9 EC E5 A0
25A5:A0 BD A0        673 MSG1      ASC   ' = '
25A8:A9 A0 BD A0     674 MSG2      ASC   ') = '
25AC:C6 F5 EE E3     675 MSG3      ASC   'Function definition'
25B0:F4 E9 EF EE
25B4:A0 E4 E5 E6
25B8:E9 EE E9 F4
25BC:E9 EF EE
25BF:C5 ED F0 F4     676 MSG4      ASC   'Empty string'
25C3:F9 A0 F3 F4
25C7:F2 E9 EE E7
25CB:          25CB  677 MSG5      EQU   *
25CB:          25CB  678 LAST      EQU   *
25CB:                679 ;ype of variable (Ø-3)
257D:                662 ;                              real,string,funct,int
257D:          0002  663 NXTARRY   DS    2              ;Beginning of array
257F:                664 ;                              after the current one
257F:          0001  665 DIM       DS    1              ;Number of dimensions
2580:                666 ;                              in array
2580:          0001  667 DIMPTR    DS    1              ;Pointer to print size
2581:                668 ;                              of each dimension
2581:A0 A4 A0 A5     669 VARSYMB   ASC   .              $ %. ;Symbols for
2585:                670 ;                              the four variable types
2585:07 C4 C9 D3     671 CMD       STR   'DISPLAY'
2589:D0 CC C1 D9
258D:D5 EE E1 E2     672 MSGØ      ASC   'Unable to read the file '
2591:EC E5 A0 F4
2595:EF A0 F2 E5
2599:E1 E4 A0 F4
259D:E8 E5 A0 E6
25A1:E9 EC E5 A0
25A5:A0 BD A0        673 MSG1      ASC   ' = '
25A8:A9 A0 BD A0     674 MSG2      ASC   ') = '
25AC:C6 F5 EE E3     675 MSG3      ASC   'Function definition'
25B0:F4 E9 EF EE
25B4:A0 E4 E5 E6
25B8:E9 EE E9 F4
25BC:E9 EF EE
25BF:C5 ED F0 F4     676 MSG4      ASC   'Empty string'
25C3:F9 A0 F3 F4
25C7:F2 E9 EE E7
25CB:          25CB  677 MSG5      EQU   *
25CB:          25CB  678 LAST      EQU   *
25CB:                679 ;
```

END OF LISTING 1

```
        KEY PERFECT 5.Ø
          RUN ON
         DISPLAY
==================================
CODE-5.Ø   ADDR# - ADDR#   CODE-4.Ø
--------   -------------   --------
1CB5Ø52D     2ØØØ - 2Ø4F     275D
E661E459     2Ø5Ø - 2Ø9F     2766
936EC55D     2ØA0 - 2ØEF     29D1
C6CA3Ø35     2ØF0 - 213F     28B6
7E74B548     2140 - 218F     2321
Ø54BD459     2190 - 21DF     21C3
C3BEA3A1     21E0 - 222F     25C3
```

```
1Ø5721AØ     223Ø - 227F     2B8Ø
B67ØBC7D     228Ø - 22CF     2AØF
7AE85E1D     22DØ - 231F     2769
54D9FE72     232Ø - 236F     299D
C1139482     237Ø - 23BF     2932
6FC52FBB     23CØ - 240F     2AE9
821AØ4Ø5     241Ø - 245F     2711
3156FEFØ     246Ø - 24AF     29EA
4113E97D     24BØ - 24FF     2615
8D8ØDEE8     25ØØ - 254F     24E7
A7FB13ØØ     255Ø - 259F     2AF3
EC3AD91Ø     25AØ - 25CA     164F
24841BØE = PROGRAM TOTAL =   Ø5CB
```

GAMES!

Reviewed by John DiPrete

Baron

With $35,000 in hand, could you become a real estate baron? This simulation challenges you to master the world of buying and selling land. In Baron, you negotiate second mortgages, put together creative financing, buy residential and business properties and speculate on land. The 37-page manual offers directions on how to play, a general introduction to real estate, helpful hints, a quick reference section, loading instructions, a glossary and an index.

You start the game with $35,000 in seed money. If you earn $1,000,000 in five years (accelerated in computer time), you achieve Baron status. Other ranks to shoot for are Investor ($100,000) and Broker ($250,000). You can save and restore a game or generate a new game each time you play, and record up to 14 previous scores. You view newspaper headlines and study charts to make investment decisions, and advance the date at monthly intervals. Monthly readouts over the five-year span reveal your capital investment outcome.

Success in the world of Baron may not guarantee similar success in the real world, but the game's portrayal of real estate transactions should give future investors a sense of what goes on. I found the experience of investing to be fascinating, but I don't recommend the program for the mathematically squeamish. You must be able to foresee trends, stay calm and patient, and master an abundance of raw data. It takes plenty of concentrated effort to master Baron.

The principal task of Baron — reading and comparing a wealth of financial data on many different charts — requires selecting from various menus, zooming in to a screen to view a chart, switching to another screen, selecting another chart to study and so on. The program responds to your keyboard input promptly, making elaborate cross-checking of information both easy and convenient.

I recommend this game to professional people or business students who are serious, painstaking and love to work for their fun.

Baron, Blue Chip Software, 6740 Eton Ave., Canoga Park, CA 91303. Type of game: Real estate simulation. Originality: Above average. Input: Keyboard. Price: $39.95.

I, Damiano

The time and place: fourteenth century Europe. The premise: You are a wizard in search of the Sacred Stone. It has the power to ward off the evil Pardo, whose armies threaten to ravage your home city of Partestrada.

As in most typical adventure games, you must solve a variety of problems to achieve your goal. You type in commands, and the computer screen reveals the outcomes of your decisions in both text and graphics. The pictures and narration appear side by side, with the graphics occupying about a third of the screen. (By the way, the screen illustrations depicted on the program package are proportionately larger than the actual screen displays.) The colorful screen imagery is adequate, but generally fuzzy looking. The text offers a vivid description of events and scenarios, but tends, at times, to be verbose.

Since you die quite easily in this game, I found the save-and-

John DiPrete, 45 Vale Ave., Cranston, RI 02910

Whether you like arcade action, educational fun, or stimulating simulations, you'll appreciate these capsule reviews of seven new games.

USER VIEWS

restore feature especially useful. Typing SAVE at frequent intervals during your adventures allows you to reenter at an advanced point in the story, should you be suddenly struck down.

Based on the fantasy trilogy written by R.A. MacAvoy, the adventure includes several characters besides Damiano — the witch, Saara; an archangel named Raphael; your faithful mutt, Macchiata; and Beelzebub. The object of the game is to achieve your goal through peaceful means. If your actions are too evil or aggressive, your good-evil soul index (displayed on the screen) will reflect this. Alas, if you behave too badly, the Evil One may claim you! In this manner, I, Damiano tends to discourage you from doing wrong — a nice touch.

Nevertheless, the satanic elements in this adventure (Lucifer, himself, plays a dominant role) will be unwelcome to the growing number of critics of games like Dungeons and Dragons. In addition, I'm sure that many who play this game will object to passages like the text that describes creatures that "drink blood through the corpses' mouths."

I have other reservations about this program. The program often prints identical responses to the player's separate, dissimilar actions. As a result, illogical events occur. For instance, when I met the hut keeper and he asked me who I was, I told him. Later, I punched him, and he knocked me out. As I awoke and tried to leave, he again asked me who I was. Was he forgetful? A nerd? Or what?

Better games of this type exist. I generally prefer picture-and-text adventures to text-only formats, but found I, Damiano weak in both areas. Its imagery is compressed and fuzzy, and its text is illogical and occasionally gruesome. I'd rate this one a C, at best.

I, Damiano, Bantam Software, 666 Fifth Ave., New York, NY 10103. Type of game: Adventure. Originality: Below average. Input: Keyboard. Price: $39.95.

Station Five

Station Five consists of four different arcade scenarios. Faced with a meteor storm, you must direct your robot, Hoofer, to save your lunar station by completing four separate tasks: beefing up the nuclear generators, building transmission towers, transporting you to the Orbiting Relay Station and supplying power to the Earth. You play against the computer at three difficulty levels using the joystick or keyboard to control Hoofer. (I preferred joystick control.)

In the first scenario, you move Hoofer on a grid of 32 squares, jumping to avoid falling meteors and tossing dead meteors into a slot to empower the particle shield. Hoofer must open and close vents to keep your reactors operating at peak efficiency — if too many are closed at one time, you blow up! The second scenario is played on another grid in a dodge-and-gather mission. In this scene, Hoofer must collect hammers (an act that automatically erects the transmission towers in the background), while simultaneously eluding "homers" and neutralizers.

In the third scenario, the pattern of jumping movement (on another 32-square grid) resembles Q-Bert's movement in the arcade game of the same name. Hoofer must jump on each Off switch to activate the transporter. But jumping on the On switches turns them off! It's an odd, invigorating setup. In the last scenario, Hoofer must collect spare parts and float or bounce toward various space platforms, while avoiding fast-moving, free-floating hazards. The arcade game Joust immediately jumps to mind.

To sum up: the detailed graphics and animation in these four-games-in-one make an interesting, fast-paced arcade experience. As for its novelty, you've already seen similar action in games like Donkey Kong, Q-Bert, Joust and others. Station Five continues a successful (though somewhat fading) tradition.

Station Five, MicroLab, Inc., 2699 Skokie Valley Rd., Highland Park, IL 60035. Type of game: Arcade. Originality: Moderate. Input: Joystick or keyboard. Price: $20.00.

Wishbringer

Interactive fiction, the relatively recent concept of involving participants directly in the plot of a story, has mushroomed into a growing number of books, games and computer programs. Wishbringer, another adventure game in the Infocom series of interactive fiction, is on a par with other Infocom classics such as the Zork series, Suspended, Starcross, Sorcerer and The Hitchhiker's Guide to the Galaxy.

The program responds to your keyboard commands. Like other Infocom adventure games, Wishbringer has a full-sentence parser. It interprets your keyboard commands with a vocabulary exceeding 1,000 words.

You start as a postal clerk on a simple errand, then find yourself in a world of magic, mayhem and mystery. To find a lost cat, you must solve puzzles, hoodwink creepies and figure out the best paths. 'Tain't easy! Patience, resolve and cleverness are essential.

One key to victory is in manipulating objects you find along your journey. You must decide what to take and what to leave behind. One fabulous little item, the Wishing-stone, enables you to cast spells (under proper conditions) for such things as good fortune, seeing ahead, camouflage, escape — and even rain. Packaged with the game disk is a real Wishing-stone, a small white rock that glows in the dark. Nice touch, Infocom people.

You may want to play this adventure game with a friend since two heads may be necessary to solve some of the knottier dilemmas. The game may take several days or weeks to solve, and you can save a game in progress on disk.

Wishbringer, Infocom, Inc., 125 Cambridge Park Dr., Cambridge, MA 02140. Type of game: Interactive fiction (fantasy), introductory level. Originality: Moderate. Input: Keyboard. Price: $39.95.

Felony!

The graphics, hint booklet, crime manual, and other paraphernalia associated with Felony! resemble another fine mystery game from CBS Software, Murder by the Dozen (reviewed in Vol. 5/No. 10).

You must solve a crime by traveling around, interviewing witnesses and suspects, hunting for clues, taking notes and forming conclusions. Collecting raw data and tying it together isn't easy — but it's fun! You must use logic and deduction to form hypotheses and test them.

Up to four people can play. The player who completes his or her investigatory task quickest is awarded the next turn. When the case has been solved, players are awarded sleuth ratings. There are 12 ratings, ranging from Clairvoyant Cop to Total Turkey. Felony! includes 12 crime cases, all at the same level of complexity.

The way both Felony! and its companion game, Murder by the Dozen, are played resembles the board classic, Clue!, except that playing on a computer greatly enhances the enjoyment. The key to winning? Take copious notes, rely on your reasoning powers, and most vital of all, consult the lab frequently for analyses of the physical evidence you've gathered.

If you'd like to be Lieutenant Columbo, temporarily at least, buy Felony!

NIBBLE LISTS

From time to time NIBBLE Magazine makes its subscriber mailing list available to reputable software and hardware manufacturers who wish to provide our readers with flyers, catalogs or other material of interest. All mailings are evaluated by NIBBLE Magazine to ensure the integrity and suitability of the material. Our policy forbids telephone or other direct solicitation by these firms, but does release your NIBBLE mailing label to them for their one-time use.

If you would prefer not to receive these mailings please send us a recent mailing label from the front of your magazine or copy the label exactly as it appears. Please send the label with a brief note directing us not to make your name available, and we will code your name for exclusion.

Felony!, CBS Software, One Fawcett Pl., Greenwich, CT 06836. Type of game: Mystery whodunit. Originality: Moderate. Input: Keyboard. Price: $24.95.

SunDog

The outstanding graphics in this adventure game deserve galactic applause. Sharp, realistic pictures of space ship features, aerial and ground views, and cities and planet systems, along with excellent close-up views, give SunDog a very sophisticated look.

At the start of each game, you must choose character traits for your hero: varying portions of intelligence, charisma, luck, strength and dexterity. Then, after setting off aboard your star freighter, you must eventually build a colony, locate frozen colonists, shoot down space hazards, trade and bargain for goods, and much more. You have a choice of wide-ranging scenarios, each with a different setup, rules and goals. You play against the Apple, and can save up to four games in progress on disk.

This isn't just another adventure game. Perhaps SunDog's finest feature is the number of different perspectives offered during the game. A player must employ a variety of distinct, creative coping skills to succeed. You can assume the role of navigator, piloting your ship and fighting off other ships; you can play a shoot-'em-up game on the ground or in space; you can become a financial investor and wheeler-dealer — and best of all, you can solve problems. Exciting and diverse elements make this a fantastic action/strategy adventure.

SunDog, FTL Games, 7907 Ostrow St., Suite F, San Diego, CA 92111. Type of game: Adventure. Originality: Moderate. Input: Two-button joystick. Price: $39.95.

Willy Byte in the Digital Dimension

Willy Byte in the Digital Dimension is an instructional arcade game that guides you through the major parts of your computer: the central processing unit, keyboard, RAM, disk drive, power source and clock. During the colorful journey, you learn about such concepts as flowcharting and how the computer works.

Using a joystick or the keyboard, you control the character Willy Byte, moving him in four directions and making him jump, pick up or discard a tool, and stop. Willy must perform several duties while avoiding the hazards of the evil Hex Luthor. Each new section offers a different scenario, with novel tasks and dangers.

The game features good graphics and nice animation, but jumping around obstacles is nothing new for the arcade veteran. You play against the Apple, and the difficulty level depends on the length of your initial message, which you must send through the various sections.

The emphasis on learning turns this game into hard work. The teaching approach seems contrived — as if to spoil one's fun. Cartoon characters can be effective aids in teaching basic concepts like the alphabet (a good example is the Charlie Brown gang in the Peanuts learning series from Random House Software). However, the Willy Byte character seems totally incongruous for introducing concepts like the relationship between an ASCII character, its hexadecimal code and its binary representation. The sprite-like Willy Byte doesn't make the material easier to digest. Quite simply, Willy just gets in the way.

Willy Byte in the Digital Dimension, Data Trek Inc., 621 Second St., Encinitas, CA 92024. Type of game: Instructional arcade. Originality: Moderate. Input: Joystick or keyboard. Price: $39.95.

Do you tend to exhibit a tendency to make punctuation errors?

Regardless of your skills,

Irregardless of your typographical errors occur.

Therefore Sensible Grammar

There fore

SENSIBLE GRAMMAR™

Yes, just like your old English teacher, your Apple Computer can now check your papers for grammatical and many other writing errors. Don't be embarrassed or lose credibility because of simple typos and mistakes. And it's so easy and fast to do with **Sensible Grammar.**

SENSIBLE GRAMMAR

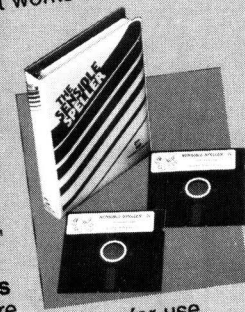
checks your grammar, capitalization, punctuation and abbreviations. It searches out phrases that are inconcise, vague, wordy or repetitive, as well as faulty, informal, pompous or sexist. It calls your attention to cliches and slang. It even allows you to enter your own personal trite and pet expressions and then lets you know every time you use them.

It Never Overrides Your Judgment

Sensibie Grammar singles out possible mistakes or improper usage and then suggests a correct replacement word or phrase, but it always leaves you the option to accept or reject the suggestion.

A Perfect Complement

SENSIBLE SPELLER™ catches your spelling mistakes, utilizing its huge 80,000 word vocabulary derived from the official Random House Dictionary®. Like **Sensible Grammar**, it works with tireless efficiency. It displays misspelled words in context, suggests the correct spelling and allows immediate replacement of misspelled words with correct ones.



Black's Law Dictionary,™ Sensible Technical Dictionary™ and **Stedman's Medical Dictionary.**™ are also available separately on diskette for use with the Sensible Speller.

AppleWorks Compatible!

*Sensible Grammar and Speller ProDOS work with the following word processors, AppleWorks, AppleWriter-ProDOS version (Apple Computer, Inc.); Format II Enhanced-ProDOS (Kensington Microware), MouseWrite (Roger Wagner Publishing); Word Juggler (Quark Inc.); Zardax-ProDOS (Computer Solutions) and others. **Sensible Speller ProDOS** also works with PFS:WRITE-ProDOS (Software Publishing, Inc.); MouseWord (International Solutions); Word-Talk (Computer Aids) and WordPerfect (SSI Software). **Sensible Speller IV** works with DOS 3.2 and DOS 3.3-AppleWriter (Apple Computers, Inc.); Bank Street Writer (Brodebund); Format II (Kensington Microware); HomeWord & Screen Writer (Sierra On-Line Inc.); †PFS: WRITE-DOS (Software Publishing, Inc.); †CP/M-Wordstar (Digital Research Corp.-Micropro International); and others. Owners of trademarks indicated in parentheses. Black's Law Dictionary (West Publishing, Inc.); Stedman's Medical Dictionary (Waverly Press, Inc.).

†Note: CP/M, Pascal and PFS:WRITE-DOS versions do not suggest correct spelling of words or allow immediate correction of misspellings.

Sensible Grammar™ is available for $99.95; **Sensible Speller**™ for $125.00 in either the "IV" or ProDOS version; **Sensible Technical Dictionary**™ for $59.95; **Stedman's Medical Dictionary**™ for $99.95; and **Black's Law Dictionary**™ for $99.95. The three specialized dictionaries require the **Sensible Speller.** All programs are available separately and run on Apple® computers.

Sensible Software,® Inc.

210 S. Woodward, Suite 229, Birmingham, MI 48011 • (313) 258-5566

CIRCLE NUMBER 41

PART V
FIRST COUSINS ONCE REMOVED

NIBBLING AT ASSEMBLY LANGUAGE

by S. Scott Zimmerman

Learn about 65C02 addressing modes and how to use them. Sample programs demonstrate implementing arrays, printing messages to the screen, using ROM routines, and using the ampersand vector.

The annual Zimmerman reunion brings together a large collection of closely-related people with a common goal — to bore each other. Along with boredom, I have the problem of trying to remember names. I can barely remember the names of my own seven children, let alone the names of my brother's and sister's kids, my cousins' kids, and my aunt's cousin's daughter's kids.

Computer programs can be much like a family reunion, with long, boring lists of data and names of variables. Like kids at the reunion, variables are difficult to manage, and their names are easy to forget.

But programmers know the secret to keeping track of closely-related data: arrays. Take, for example, the program RAIN .BASIC (**Listing 1**). Type the program in and SAVE it to disk with:

```
SAVE RAIN.BASIC
```

When you run the program, 18 raindrops fall from the top of the screen to the bottom. This simple program illustrates the use of arrays. The program keeps track of the locations of 18 raindrops and moves them down the screen, using three one-dimensional arrays. (Alternatively, the program could have used one three-dimensional array.)

The program initializes the XR and YR arrays in **line 110** by reading the data found in **line 190**, and it plots the raindrops in their initial locations. **Lines 120-170** form the main program loop, in which raindrops move down the screen. The program uses the RF (rain flag) array to determine if a raindrop is still on the screen. Once a raindrop falls below position YR = 159 (see **line 150**), the program erases it from the screen and decrements the raindrop counter, N. The program ends when N reaches zero, indicating that no raindrops remain on the screen.

ASSEMBLY LANGUAGE ARRAYS

Even with little programming experience, you can grasp the usefulness of arrays by analyzing RAIN.BASIC. Arrays in machine language are just as important.

To illustrate arrays in machine language, I have written RAIN.ML, the machine language version of RAIN.BASIC (**Listing 2**). Key the source program into your assembler, assemble the program, and save the object code as RAIN.ML. To run the program, type:

```
BRUN RAIN.ML
```

or type:

```
BLOAD RAIN.ML
```

followed by:

```
CALL 16384
```

I will explain the program in detail later, but first, let's look at **lines 40-47**, in which the array RAINFLAG is initialized to zero. Initialization of arrays is often unnecessary in BASIC, since all variables, including arrays, are automatically set to zero when you RUN a program. But the contents of the region of memory designated for the RAINFLAG array in the machine language program are unknown, and therefore you must initialize the array values explicitly.

To initialize the array RF( ) to zero in BASIC, you would write:

```
FOR I=1 TO N: RF(I) = 0: NEXT I
```

where N is the number of elements in the array and I is the index. RAIN.ML, on the other hand, uses the X-Register as the index. The program sets the X-Register to NUMDOTS in **line 43**. In Part IV of this series (Vol. 6/No. 11) machine language programs were used to execute loops in reverse order, starting at the highest index value (in this case, NUMDOTS) and proceeding down to zero. In **line 44**, the program sets the Accumulator to zero, and the loop actually begins in **line 45**.

Here is the key point: In **line 45** the command STA RAINFLAG,X stores the contents of the Accumulator into RAINFLAG plus X, not into RAINFLAG ($40DB). If

by S. Scott Zimmerman, 1129 East 470 North, Orem, UT 84057. The programs run under DOS 3.3 and ProDOS.

the X-Register happens to contain two, for example, then RAINFLAG,X represents the address $40DB+2, or $40DD. In this case, the X-Register is an index; in fact, the X-Register is often called the X index register. The operand, RAINFLAG,X, specifies the memory location of element X of the RAINFLAG array.

ADDRESSING MODES

The command STA RAINFLAG,X is an example of the absolute indexed addressing mode. An addressing mode is the way that an instruction (such as LDA, STA, LDY, or STY) addresses memory.

If you have followed the Nibbling at Assembly Language series, you have already used several different addressing modes. Consider the following uses of the LDA (Load Accumulator) instructions:

```
LDA #10      ;Load value 10 ($0A) into Accumulator
LDA YVAL  ;Load value found at YVAL
```

The command LDA #10 is an example of the *immediate* addressing mode. The Accumulator is loaded with the byte value (in this case 10 or $0A), which immediately follows the opcode in memory. The object code for LDA #10 is A9 0A.

The command LDA YVAL (see **line 86** of **Listing 2**) is an example of the absolute addressing mode. The Accumulator is loaded with the byte value currently at address YVAL. Since the address of YVAL is $40B6 in RAIN.ML, LDA YVAL loads the Accumulator with the byte value currently stored at $40B6. The object code for LDA YVAL is AD B6 40. This is called the absolute addressing mode because the operand address is used without modification. As you will see with other addressing modes, the operand may indicate a relative address, or it may indicate where to find another address.

Table 1 lists the addressing modes available with the 65C02 microprocessor (used in the //c and enhanced //e). Except where noted, the syntax is the same for the 6502 microprocessor (used in the II Plus and //e). I will not discuss all of these addressing modes at this time, but will cover them as needed throughout the series.

Not all of these addressing modes can be used by all of the opcodes. For instance, LDA can use immediate, absolute, zero page, absolute indexed by X or Y, zero page indexed by X or Y, indirect indexed, indexed indirect and zero page indirect addressing modes, while BNE uses only relative addressing mode. For a complete list of opcodes with their allowed addressing modes, consult your *Apple // Reference Manual*.

COMPARING THE LISTINGS

By comparing RAIN.BASIC (**Listing 1**) and RAIN.ML (**Listing 2**), you can readily see some of the differences between BASIC and assembly language. Although the assembly source code listing is much longer, the object code (the machine language program) is only 273 bytes, compared to 347 bytes for the BASIC program (excluding the REM statements in **lines 10-80**). In addition, RAIN.BASIC uses 20 bytes for simple variables and 307 bytes for arrays, whereas the machine language program uses no additional bytes for variables and arrays. The total memory requirements are 674 bytes for the BASIC program and 273 bytes for the machine language program.

RAIN.ML not only uses less memory, but, as you would expect, it is much faster: RAIN.BASIC takes 93.7 seconds to run, while RAIN.ML takes only 2.3 seconds. Although programming in assembly language is more difficult than in BASIC, your efforts are rewarded by faster, more compact programs.

HOW RAIN.ML WORKS

RAIN.ML demonstrates the use of arrays and several addressing modes. The array containing the initial X,Y coordinate values of raindrops is called XYSTART (**lines 128-131** of **Listing 2**). **Lines 51-70** plot the raindrops in their initial locations on the Hi-Res screen (analogous to **line 140** of RAIN .BASIC). **Line 51** initializes the loop index to zero, using the immediate addressing mode to load the X-Register. The array index, which in this case is different from the loop index, is set to zero in **line 52**, again using the immediate addressing mode.

The loop that plots the raindrops starts in **line 53**. The values of the loop and array indexes are stored in locations LOOPINDX and ARRINDX, using the absolute addressing mode. The initial X-coordinate of the current raindrop is loaded into the Accumulator in **line 55**, using the absolute indexed addressing mode. For example, when the loop index is zero, element 0 of the XYSTART array, located at $40ED, is accessed; when the loop index is one, element 1 of the XYSTART array, located at $40EE, is accessed; and so forth.

Note that in assembly language, you nearly always use the zeroth element of an array. Even though element 0 in BASIC and Pascal arrays nearly always exists, it is seldom actually used.

Line 56 of RAIN.ML stores the X-coordinate from the XYSTART array into the XRAIN array, whose index is located in the Y-Register. This is another example of the absolute indexed addressing mode, but this time using the Y-Register as the index.

Line 57 saves the X-coordinate in the variable XVAL, using the absolute addressing mode. The INX instruction in **line 58** increments the X-Register (which contains the loop index), using the *implied* addressing mode. As the name indicates, the operand (X, in this case) is implied in the instruction itself.

Line 59 saves the new X-value in LOOPINDX, using the absolute addressing mode. **Lines 60 and 61** transfer the initial Y-coordinate value from the XYSTART array to the YRAIN array, using absolute indexed addressing modes.

Line 62 uses the absolute addressing mode to load the X-Register with XVAL, and **line 63** uses the immediate addressing mode to load the Y-Register with zero. **Line 64** uses the Applesoft ROM routine, HPLOT, to plot a point on the Hi-Res screen at the location indicated by the contents of the Accumulator and the X and Y Registers. I will explain this in more detail later in the USING ROM section.

After plotting a dot (**line 64**), the program restores the loop index and the array index into the X and Y Registers, respectively (see **lines 65 and 66**). **Lines 67 and 68** increment these two indexes, and then **line 69** checks to see if the Y-Register (the array

TABLE 1: 65C02 Addressing Modes

Addressing Mode	Example Syntax	Example Object Code
Immediate	LDA #$1F	A9 1F
Absolute	LDA $6AD5	AD D5 6A
Zero page	LDA $1F	A5 1F
Accumulator	ASL	0A
Implied	INX	E8
Absolute indexed, X	LDA $6AD5,X	BD D5 6A
Absolute indexed, Y	LDA $6AD5,Y	B9 D5 6A
Zero page indexed, X	LDA $1F,X	B5 1F
Zero page indexed, Y	LDX $1F,Y	B6 1F
Indirect indexed	LDA ($1F),Y	B1 1F
Indexed indirect	LDA ($1F,X)	A1 1F
Relative	BNE $1F	D0 1F
Indirect	JMP ($6AD5)	6C D5 6A
Zero page indirect*	LDA ($1F)	B2 1F

*Not available on the 6502

Apple II + /IIe/IIc

Nine reasons why the MicroSPARC Assembler is a red-hot bestseller.

1 EASE OF USE. Now even a beginner can enjoy all the benefits of programming in Assembly language — with this *menu-driven* Macro Assembler/Editor. No commands to memorize! Even help screens for quick and easy reference.

2 POWER. Power and *speed* make this assembler ideal for the more advanced user, whether you're writing a quick routine, or considering a full-scale software development. You'll never outgrow it!

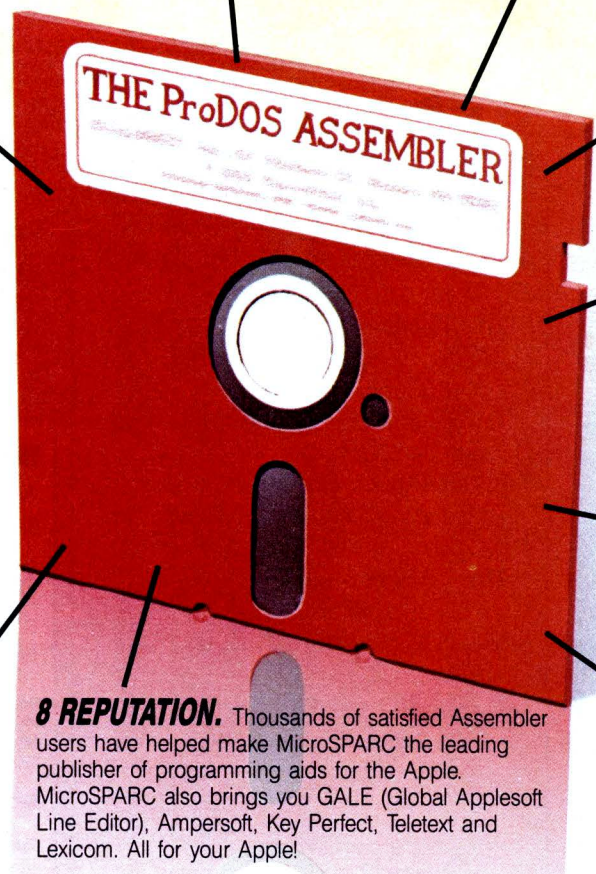
3 MacroSoft. Consider the purchase of the MicroSPARC Assembler's companion program, MacroSoft. And write Applesoft-like programs which are then converted into machine language by the Assembler — like magic!

4 ADAPTABILITY. The Assembler automatically adapts to whatever Apple you have, whether it's 40 or 80 column display, upper or lower case, or extended memory.

5 ADVANCED FEATURES. The Assembler comes complete with a powerful editor with features like Global Search and Replace, plus a library of subroutines. A collection of Macro commands even simulates the commands of other assemblers, like Applesoft Toolkit.

6 SUPERIOR DOCUMENTATION. Learn the ins and outs of using the MicroSPARC Assembler with a concise, easy-to-follow manual. A minimum of jargon, and lots of examples to make sure you understand.

7 SUPPORT. Purchase the Assembler and get a priceless extra — the MicroSPARC team of technical experts, FREE! You'll be just a phone call or letter away from help and advice, should you need assistance.



8 REPUTATION. Thousands of satisfied Assembler users have helped make MicroSPARC the leading publisher of programming aids for the Apple. MicroSPARC also brings you GALE (Global Applesoft Line Editor), Ampersoft, Key Perfect, Teletext and Lexicom. All for your Apple!

9 PRICE. We've saved the best for last: The Assembler sells for just $49.95, making it the best assembler buy available. Anywhere!

UPGRADE OFFER

Current owners of the DOS 3.3 version of the Assembler (and MacroSoft) can receive the ProDOS version(s) by sending $19.95 for the Assembler alone, or $29.95 for the ProDOS Assembler and Macrosoft, along with the cover(s) of the original documentation manual(s).

ProDOS versions are completely compatible with all existing Assembler and MacroSoft programs. New and expanded documentation is included in your upgrade.

To order, fill out the attached coupon or call (617) 371-1660. Be sure to specify the DOS 3.3 or the ProDOS version.

☐ **I want the Assembler! Here's my $49.95.**

☐ **I want the Assembler and MacroSoft! Here's my $99.95**

PLEASE CHOOSE ONE:
☐ DOS 3.3 Version
☐ ProDOS Version

***UPGRADE OFFER**
I want to upgrade my current DOS 3.3 version(s) to ProDOS. I've enclosed the cover page(s) from my original documentation manual(s).

☐ $19.95 / Assembler
☐ $29.95 / Assembler & MacroSoft

Name _____
Address _____
City _____ State _____ Zip _____

(POSTPAID)
☐ MasterCard ☐ Visa ☐ Check, M.O. (Mass. residents add 5% sales tax)
Charge Card # _____ Expiration Date _____
Signature _____ Tel. # _____
Payable in U.S. funds only.

MicroSPARC Inc. **45 Winthrop St., Concord, MA 01742**

CIRCLE NUMBER 42

NIBBLE EXPRESS IV

The $500 Program Book for your Apple

That's right! More than $500 worth of programs for your Home, Business, Education and Entertainment. NIBBLE EXPRESS IV is an Anthology of the Major Articles and Programs appearing in Volume 4 (1983) of NIBBLE Magazine.

NIBBLE EXPRESS IV contains *up-to-date enhancements* and *complete program listings* with comprehensive *articles* that show:

What each program does.

How to type them into your Apple, ACE, or other Applesoft-compatible computer.

How to use them.

How to customize them to your own needs.

Here's a representative sample of the 50 programs and articles:

The Investor—*Stock Tracking, Reporting, and Graphics.*
The Librarian—*Automatic control of your Disk Libraries.*
The Designer/Illustrator—*Art Design/Creation and Composition.*
RAM Manager—*Tuck away up to 10 Programs in RAM Card Memory.*
The Programmer—*Applesoft Preprocessor for Structured BASIC.*
T.U.N.E.S.—*Music Composition and Playback.*
RAM Disk—*Make a Phantom Disk out of your 16K RAM Card.*
The Nibbler—*Arcade Action with the famous Pellet Eater.*
Math Monster—*Arithmetic can be fun.*
Personal Appointment Calendar—*Track events, dates, and meetings.*

AND THERE'S MORE...

The NIBBLE EXPRESS is full of useful Tips 'N Techniques, Utilities, and other Goodies for getting more out of your Apple.

Even if you own all 8 issues of NIBBLE in 1983, you'll want the EXPRESS IV to have updated program listings in one convenient package. It's a MUST for your library.

nibble 45 Winthrop, St., Concord, MA 01742

Yes I want NIBBLE EXPRESS in my library!

Send me:

- ☐ Nibble Express Volume I at $14.95 each.
- ☐ Nibble Express Volume II at $14.95 each.
- ☐ Nibble Express Volume III at $17.95 each.
- ☐ **Nibble Express Volume IV at $17.95 each.**
- ☐ Nibble Express Volume V at $19.95 each.

Please add $1.75 shipping/handling per copy (outside U.S. add: $2.75 surface or $6.50 Air Mail per copy). Mass. residents add 5% sales tax. Payable in U.S. funds only.

Enclosed is my: ☐ Check ☐ Money Order

Bill my: ☐ MasterCard ☐ VISA

Card # _____

Expires _____ Phone _____

Signature _____

Name _____

Street _____

City _____

State _____ Zip _____

Your check or money order must accompany your order to qualify. Outside U.S.: Checks must be drawn on a U.S. bank. *Apple is a registered trademark of Apple Computer Company.

index) has reached NUMDOTS, the number of dots. If not, **line 70** makes the program branch back to PLOTLOOP in **line 53**.

You should be able to understand the assembly language code in **lines 74-113** of **Listing 2**.

ALLOCATING MEMORY

When writing a BASIC program, you seldom need to worry about how your program variables and arrays use memory. The BASIC interpreter sets up the memory locations automatically. However, when you program in assembly language, you must specify the memory location of all variables and arrays. Thus, in **lines 119-127** of **Listing 2**, RAIN.ML contains statements that reserve regions of memory for the variables NUMDOTS, NDOTSON, LOOPINDX, and so on. For example:

```
RAINFLAG DFS 18
```

The command DFS is an assembler directive (or pseudo-opcode) that tells the assembler to define storage (DFS), for example, to reserve 18 bytes of memory. The address $40DB at the start of the memory region (not actually included in **Listing 2**), becomes equated to the label RAINFLAG.

If you want to initialize an array to a set of values in BASIC, you usually use a DATA statement to store the initial array data. In assembly language, you use an assembler directive or pseudo-opcode that lets you put numbers directly into your object code. One way to do this is with the DFC (define constant) assembler directive, as shown in **lines 128-131** of **Listing 2**. Some assemblers use a different pseudo-opcode name, such as DFB (define byte) or DC (define constant). Read your assembler's manual for the correct pseudo-op name and an explanation of how to use it.

USING ROM ROUTINES IN ASSEMBLY LANGUAGE

Your Apple's ROM is full of wonderful machine language subroutines. Without them, most short, simple assembly language programs would be enormous.

RAIN.ML makes liberal use of ROM routines. For example, the routine to clear the text screen — the Applesoft HOME command — is located in ROM at address $FC58. The routine to clear and initialize Hi-Res page 1 — the Applesoft HGR command — is located in ROM at address $F3E2. These two routines are simple to use in assembly language. You just have to EQUate an appropriate name (e.g., HOME and HGR) to the address of the routine (see **lines 19-23** in **Listing 2**), and then call the routines with a JSR (see **lines 30-31**). That's all there is to it.

Other ROM routines are not so simple. They expect to find data in certain 65C02 registers. For example, the SETHCOL routine sets HCOLOR for the points you subsequently plot on the Hi-Res screen. Before doing a JSR to SETHCOLOR (at $F6EC), the desired color code (0-7) must be present in the X-Register (see **lines 32-33** in **Listing 2**).

A still more complicated ROM routine is HPLOT, found at $F457 (see **lines 20, 87 and 103** of **Listing 2**). HPLOT requires that the program specify the horizontal (or X-coordinate) and the vertical (or Y-coordinate) of the point to be plotted. This is actually not very difficult.

The X-coordinate can have a value from 0 to 279 ($00 to $117), and therefore requires two bytes. The program must load the low-order byte (LOB) value of the horizontal coordinate into the X-Register (**line 84** of **Listing 2**), and the high-order byte (HOB) into the Y-Register (**line 85**).

Since the Y-coordinate can only have a value from 0 to 191 ($00 to $BF), it requires only one byte. The program must load the byte value of the vertical position into the Accumulator. Once the horizontal and vertical values are in the 65C02 internal registers, the program does a JSR to HPLOT ($F457) to turn on the desired pixel.

ROM INFORMATION SOURCES

Many *Nibble* readers have written to me with the question, "How do I find out what ROM routines are available and how do I learn how to use them?" There are five major sources of information about ROM:

1. Many programs in *Nibble* are written in assembly language, and most of these contain Applesoft ROM or Monitor ROM references. An excellent way to learn more is to study the source code and read the explanations in these articles. More particularly, the Disassembly Lines series by Sandy Mossberg explains the ROM routines in depth. And, of course, Nibbling At Assembly Language will include future examples of ROM routines.
2. *All About Applesoft* is a must for all serious assembly language programmers. It is published by A.P.P.L.E. (Apple PugetSound Program Library Exchange), 290 S.W. 43rd Street, Renton, WA 98055. It contains detailed explanations of many Applesoft ROM routines.
3. Another valuable tool for assembly language programmers is the book *Apple II Monitors Peeled* published by Apple Computer, Inc., 10260 Brandley Drive, Cupertino, CA 95014. It contains explanations of numerous Monitor ROM routines and shows how to access them from assembly language programs.
4. Apple also publishes the *Apple // Reference Manual* and *Reference Manual Addendum: Monitor ROM Listings*. If you have them on your bookshelf, dust them off and look them over!
5. Finally, I recommend *What's Where in the Apple* by William F. Luebbert, published by Micro Ink, Inc., P.O. Box 6502, Chelmsford, MA 01824. This book lists all the Applesoft and Monitor ROM routines, although the explanations of how to use them are often too brief to be of solid value.

In addition to these resources, you may want to consult other texts on Apple assembly language.

LISTING 1: RAIN.BASIC

```
10  REM ************************
20  REM *      RAIN.BASIC      *
30  REM * BY SCOTT ZIMMERMAN *
40  REM * COPYRIGHT (C) 1986 *
50  REM * By MICROSPARC, Inc *
60  REM * CONCORD, MA  01742 *
70  REM ************************
80  REM
90  HOME : HGR : HCOLOR= 3
100 N = 18:M = N: DIM XR(N),YR(N),RF(N)
110  FOR I = 1 TO N: READ XR(I),YR(I): HPLOT
     XR(I),YR(I): NEXT I
120  FOR I = 1 TO M
130  IF RF(I) THEN 170
140  HCOLOR= 0: HPLOT XR(I),YR(I): HCOLOR= 3
150 YR(I) = YR(I) + 1: IF YR(I) > 159 THEN RF
     (I) = 1:N = N - 1: ON N = 0 GOTO 180: GOTO
     170
160  HPLOT XR(I),YR(I)
170  NEXT I: GOTO 120
180  TEXT : HOME : END
190  DATA 22,0,28,18,224,5,56,3,182,24,238,17
     ,210,20,112,14,196,14,140,42,154,5,168,2
     ,70,10,126,16,98,4,17,10,84,21,252,0
```

END OF LISTING 1

continued on page 99

ENTERING MSG.PRINT

Now let's look at a short example showing more uses of the indirect indexed addressing mode and new ROM routines. MSG.PRINT demonstrates message printing from your assembly language program.

Key MSG.PRINT (**Listing 3**) into your assembler, assemble the program, and save both the source code and object code (MSG .PRINT) to disk.

To execute MSG.PRINT, first make sure you are in 40-column mode, then **BLOAD MSG.PRINT** into your Apple, and type **CALL 16384** from BASIC or type **4000G** from the Monitor. MSG.PRINT clears the text screen and then prints two lines of text.

HOW MSG.PRINT WORKS

I have arbitrarily selected $4000 as the starting address of MSG.PRINT (see **line 15** of **Listing 3**). I could have selected $0300 (the page 3 user's space) or $9400 (near DOS or ProDOS) or any other free space.

In the EQUates section of MSG.PRINT (**lines 21-25**), MSGPTR is a special kind of variable called a pointer. Pointers are two-byte zero page locations that contain a two-byte address in reverse byte order, LOB (low-order byte) first and HOB (high-order byte) second. For example, if the two bytes at MSGPTR ($00) and MSGPTR+1 ($01) contained $30 and $40, respectively, we would say that MSGPTR points to the address $4030.

I arbitrarily selected $00 and $01 as the address of MSGPTR. I could have selected any two-byte zero page sequence that does not interfere with DOS or BASIC usage. Safe addresses for your assembly language programs include $00-$09, $1A-$1C and $1E-$1F. If you are not sure how safe a zero page address is, consult your *Apple // Reference Manual* or *What's Where In The Apple*.

The other four symbols in the EQUates section are system addresses used by the Monitor ROM. CH is the zero page location that contains the horizontal position of the screen cursor. It uses values 0-39 ($00 to $27) in 40-column mode or 0-79 ($00 to $4F) in 80-column mode. TABV is the Monitor routine for setting the cursor at a specified location on the screen. HOME is the Monitor routine for clearing the text screen and moving the cursor to the upper-left corner. COUT, one of the most frequently used Monitor ROM routines, outputs a character to the screen. In addition, it handles screen scrolling and automatically updates the internal cursor values as each character is printed, so that the next character goes to the correct location.

Now let's go through the program itself to see how these Monitor ROM routines are used and to learn more about addressing modes. **Line 31** clears the text screen. **Lines 32-35** demonstrate a handy sequence of commands that set the cursor to any location on the text screen. To do this, load the

continued on page 103

LISTING 2: RAIN.ML

```
0                       :
1                       ********************************************
2                       *                                          *
3                       *                RAIN.ML                   *
4                       *                                          *
5                       *           By S. Scott Zimmerman          *
6                       *                                          *
7                       *            Copyright (c) 1986            *
8                       *            by MicroSPARC, Inc            *
9                       *            Concord, MA  01742            *
10                      *                                          *
11                      ********************************************
12
13                              ORG $4000       ;Put above hi-res page
14
15                      ********************************************
16                      *  EQUATES:                                *
17                      ********************************************
18
19                      HGR     EQU $F3E2       ;Basic HGR command
20                      HPLOT   EQU $F457       ;Basic HPLOT command
21                      SETHCOL EQU $F6EC       ;Basic HCOLOR command
22                      SETTEXT EQU $FB39       ;Basic TEXT command
23                      HOME    EQU $FC58       ;Basic HOME command
24
25                      ********************************************
26                      *  PROGRAM:                                *
27                      ********************************************
28
29
30  4000  20 58 FC              JSR HOME        ;Clear text screen
31  4003  20 E2 F3              JSR HGR         ;Init graphics
32  4006  A2 03                 LDX #3          ;3 = white
33  4008  20 EC F6              JSR SETHCOL     ;Set hi-res color
34
35                      * BASIC LINE 100:
36
37  400B  A9 12                 LDA #18         ;Set the number of dots
38  400D  8D B1 40              STA NUMDOTS
39  4010  8D B2 40              STA NDOTSON     ;Set number dots "on"
40
41                      * INITIALIZE FLAG ARRAY:
42
43  4013  AE B1 40              LDX NUMDOTS     ;Set the index
44  4016  A9 00                 LDA #0
45  4018  9D DB 40  INITLOOP    STA RAINFLAG,X  ;Zero the index
46  401B  CA                    DEX             ;Next index
47  401C  10 FA                 BPL INITLOOP    ;Not done, continue
48
49                      * BASIC LINE 110:
50
51  401E  A2 00                 LDX #0          ;Init the loop index
52  4020  A0 00                 LDY #0          ;Init the array index
53  4022  8E B3 40  PLOTLOOP    STX LOOPINDX    ;Save loop index
54  4025  8C B4 40              STY ARRINDX     ;Save array index
55  4028  BD ED 40              LDA XYSTART,X   ;Set X array
56  402B  99 B7 40              STA XRAIN,Y     ; to initial value
57  402E  8D B5 40              STA XVAL        ;Save current XVAL
58  4031  E8                    INX             ;Go to next index
59  4032  8E B3 40              STX LOOPINDX    ;Save it
60  4035  BD ED 40              LDA XYSTART,X   ;Set Y array
61  4038  99 C9 40              STA YRAIN,Y
62  403B  AE B5 40              LDX XVAL        ;Restore XVAL
63  403E  A0 00                 LDY #0          ;Zero the HOB
64  4040  20 57 F4              JSR HPLOT       ;Plot the dot
65  4043  AE B3 40              LDX LOOPINDX    ;Restore
66  4046  AC B4 40              LDY ARRINDX     ;Restore
67  4049  E8                    INX             ;Go to next loop index
68  404A  C8                    INY             ;Go to next array index
69  404B  CC B1 40              CPY NUMDOTS     ;Up to num dots?
70  404E  90 D2                 BCC PLOTLOOP    ;No, so continue
71
72                      * BASIC LINE 120-160:
73
74  4050  AE B1 40  STARTMOV    LDX NUMDOTS     ;Set loop
75  4053  8E B3 40  MOVLOOP     STX LOOPINDX    ;Save the index
76  4056  BD DB 40              LDA RAINFLAG,X  ;Is this on?
77  4059  D0 46                 BNE ENDLOOP     ;Not on, so end loop
78  405B  BD B7 40              LDA XRAIN,X     ;Get X value
79  405E  8D B5 40              STA XVAL
80  4061  BD C9 40              LDA YRAIN,X     ;Get Y value
81  4064  8D B6 40              STA YVAL
82  4067  A2 00                 LDX #0          ;Zero color
83  4069  20 EC F6              JSR SETHCOL
84  406C  AE B5 40              LDX XVAL        ;Get X value
85  406F  A0 00                 LDY #0          ;Zero the HOB
86  4071  AD B6 40              LDA YVAL        ;Get Y value
87  4074  20 57 F4              JSR HPLOT       ;Erase the point
88  4077  A2 03                 LDX #3          ;Restore color
89  4079  20 EC F6              JSR SETHCOL     ; to white
90  407C  AE B3 40              LDX LOOPINDX    ;Restore loop index
91  407F  FE C9 40              INC YRAIN,X     ;Go down screen one
92  4082  BD C9 40              LDA YRAIN,X     ;Get new value
93  4085  C9 9F                 CMP #159        ;Down at bottom?
94  4087  90 10                 BCC CONTLOOP    ;No, continue loop
95  4089  A9 01                 LDA #1          ;Set the flag
96  408B  9D DB 40              STA RAINFLAG,X
```

continued on next page

LISTING 2: RAIN.ML *(continued)*

```
  97  408E  CE B2 40          DEC NDOTSON     ;Decrease num dots
  98  4091  AD B2 40          LDA NDOTSON     ;Are they all gone?
  99  4094  F0 14             BEQ QUIT        ;Yes, so quit
 100  4096  4C A1 40          JMP ENDLOOP     ;Go to end of loop
 101  4099  AE B5 40  CONTLOOP LDX XVAL        ;Get same X value
 102  409C  A0 00             LDY #0          ;Make sure HOB is 0
 103  409E  20 57 F4          JSR HPLOT       ;Plot new point
 104  40A1  AE B3 40  ENDLOOP  LDX LOOPINDX    ;Restore loop index
 105  40A4  CA                DEX             ;Go to next index
 106  40A5  10 AC             BPL MOVLOOP     ;Not done
 107  40A7  4C 50 40          JMP STARTMOV    ;Jump to start
 108
 109                  * BASIC LINE 170:
 110
 111  40AA  20 39 FB  QUIT     JSR SETTEXT     ;Go set text mode
 112  40AD  20 58 FC           JSR HOME        ;Clear screen
 113  40B0  60                 RTS
 114
 115                  ********************************************
 116                  *  VARIABLES and DATA:                     *
 117                  ********************************************
 118
 119                  NUMDOTS  DFS 1
 120                  NDOTSON  DFS 1
 121                  LOOPINDX DFS 1
 122                  ARRINDX  DFS 1
 123                  XVAL     DFS 1
 124                  YVAL     DFS 1
 125                  XRAIN    DFS 18
 126                  YRAIN    DFS 18
 127                  RAINFLAG DFS 18
 128  40ED  16 00 1C  XYSTART  DFC 22,0,28,18,224,5,56,3,182,24,238,17
           12 E0 05 38 03 B6
           18 EE 11
 129  40F9  D2 14 70           DFC 210,20,112,14,196,14,140,42,154,5
           0E C4 0E 8C 2A 9A
           05
 130  4103  A8 02 46           DFC 168,2,70,10,126,16,98,4,17,10,84,21
           0A 7E 10 62 04 11
           0A 54 15
 131  410F  FC 00              DFC 252,0

000  Errors

4000  Hex Start of Object
4110  Hex end of Object
0111  Hex Length of Object
7B33  Hex end of Symbols
```

END OF LISTING 2

LISTING 3: MSG.PRINT

```
  0                    ;
  1                    ********************************************
  2                    *                                          *
  3                    *                MSG.PRINT                 *
  4                    *                                          *
  5                    *           By S. Scott Zimmerman          *
  6                    *                                          *
  7                    *            Copyright (c) 1986            *
  8                    *            by MicroSPARC, Inc            *
  9                    *            Concord, MA  01742            *
 10                    *                                          *
 11                    *       The MicroSPARC Assembler 3.0       *
 12                    *                                          *
 13                    ********************************************
 14
 15                           ORG $4000       ;Above hi-res screen 1
 16
 17                    ********************************************
 18                    *  EQUATES:                                *
 19                    ********************************************
 20
 21                    MSGPTR   EQU $00         ;Message pointer
 22                    CH       EQU $24         ;Horizontal cursor pos
 23                    TABV     EQU $FB5B       ;Mon ROM tab routine
 24                    HOME     EQU $FC58       ;Mon HOME routine
 25                    COUT     EQU $FDED       ;Mon ROM character out
 26
 27                    ********************************************
 28                    *  PROGRAM:                                *
 29                    ********************************************
 30
 31  4000  20 58 FC           JSR HOME        ;Clear the screen
 32  4003  A2 05              LDX #5          ;Do HTAB 5
 33  4005  86 24              STX CH          ;Store in mon HTAB loc
 34  4007  A9 08              LDA #8          ;Do VTAB 8
 35  4009  20 5B FB           JSR TABV        ;Call mon tabbing rtn
 36
 37  400C  A9 30              LDA #MSG1       ;Set pointer to message 1
 38  400E  85 00              STA MSGPTR
 39  4010  A9 40              LDA #MSG1/      ;Do the HOB
 40  4012  85 01              STA MSGPTR+1
 41  4014  20 23 40           JSR MSGPRINT    ;Go print the message
 42
 43  4017  A9 51              LDA #MSG2       ;Set pointer to message 2
```

continued on page 104

NEW PRODUCT

IIc SYSTEM CLOCK

- Fully ProDos compatible
- Automatic time and date stamping
- Easy to use from BASIC
- Battery operated, uses 3 "AA" batteries (will last 1-2 years before simple replacement)
- Date has year, month, date and day of week
- Time has hours, minutes and seconds
- Will time and date stamp AppleWorks files
- Will display time and date on the AppleWorks screen
- Auto access from AppleWorks data-base (just use a time and date field)
- Pass through serial port - The IIc system clock can plug into either the modem or printer serial port, then modem or printer plugs into the clock
- No hassle 5 year warranty
- Only $79.00

"We Set the Standard"

214-241-6060 APPLIED ENGINEERING 9 AM - 11 PM

If the sticker on your printer reads like this, you've been stuck.

Too many big-winded printer companies are sticking it to too many unsuspecting consumers. Making a lot of noise about features that aren't featured at all.

It's time to strike back at the stickers. And sticker shock.

THE STAR SG-10

- *2K BUFFER* ..... ***STANDARD***
- *100% IBM-compatible* ..... ***STANDARD***
- *ADJUSTABLE TRACTOR* .... ***STANDARD***
- *HEX DUMP* ..... ***STANDARD***
- *ITALICS* ..... ***STANDARD***
- *NEAR-LETTER-QUALITY* ... ***STANDARD***
- *RETAIL PRICE* ***$299***

The SG-10 printer from Star Micronics includes as standard many of the attractive features which Epson's® LX-90 and the Okidata® ML-182 charge extra for. Or don't offer at all.

The Star SG-10 is faster than the Epson and more IBM®-compatible. And unlike the Okidata, the SG-10 features dual-mode printing—draft and NLQ—at no extra charge.

The SG-10 offers more at less cost—or much more for the same price—than its two leading competitors.

So, if you'd rather be struck by a sticker that makes great sense than stuck with a sticker that makes nonsense, look into the SG-10 today. At your local Star dealer.

star® micronics

THE POWER BEHIND THE PRINTED WORD®

® Epson is a registered trademark of Epson America, Inc. ® IBM is a registered trademark of International Business Machines, Inc. ® Okidata is a registered trademark of Okidata, an OKI AMERICA company

CIRCLE NUMBER 45

desired horizontal position into CH ($24); this has the same effect as the HTAB command in BASIC. Then, load the Accumulator with the desired vertical position (the VTAB value). Finally, call TABV, which does the necessary calculations for COUT to print the next character at the specified location.

Lines 37-40 set the message pointer, MSGPTR. The first message has the label MSG1 (**line 66** of **Listing 3**). MSG1 is simply a group of bytes that contains the ASCII codes of the sentence "NIBBLING AT ASSEMBLY HAS IT ALL!" **Lines 37-38** load the LOB ($30) of MSG1 into the Accumulator and store the value into MSGPTR. **Lines 39-40** load the HOB ($40) of MSG1 into the Accumulator and store it into MSGPTR. MSGPTR now is pointing at the address of MSG1, namely, $4030.

With MSGPTR pointing at the address of the message, **line 41** calls the subroutine MSGPRINT (message print), which actually prints the message onto the screen. **Lines 43-47** reset the pointer to the second message, MSG2, and again call MSGPRINT to print the second message.

The most important part of MSG.PRINT is, of course, the subroutine MSGPRINT, **lines 54-60**. This is how it works. **Line 54** initializes the Y-Register to zero; this register serves as the index for the print loop. In **line 55**, the pointer MSGPTR loads a character into the Accumulator. This is a good example of the *indirect indexed* addressing mode. The LDA command here does not load the Accumulator with the value at MSGPTR; instead it loads the value pointed at by MSGPTR. In other words, the load is not direct (or absolute) but rather indirect.

Furthermore, LDA (MSGPTR),Y loads the Accumulator with the value pointed at by MSGPTR only when the Y-Register is zero. When the Y-Register is nonzero, the accessed memory address is indexed by the value in Y. For example, if the Y-Register contains 3 and MSGPTR is pointing at (or contains the address) $4030, the byte value at $4030+3 (that is, at $4033) is loaded into the Accumulator.

Let's compare the indirect indexed addressing mode with other more familiar addressing modes. The assembly language statement LDA MSGPTR is an example of the absolute addressing mode, in which the Accumulator is loaded with the contents at the memory address MSGPTR (i.e., address $00). The statement LDA MSGPTR,Y is an example of the absolute indexed addressing mode, in which the Accumulator is loaded with the contents at the address MSGPTR + Y. If Y contains $01, the value at MSGPTR+$01 is loaded into the Accumulator. In contrast, the statement LDA (MSGPTR), Y loads into the Accumulator the value contained at the address stored in MSGPTR and MSGPTR+1, offset by the value in the Y-

Register. Indirect indexed addressing is further illustrated in the program AUTOTAB (**Listing 4**).

In MSG.PRINT (**Listing 3**), **line 56** branches to the end of the MSGPRINT subroutine if the value loaded into the Accumulator is zero. Therefore, each message to be printed should be terminated with a zero (see **line 67**, in which the DFB assembler instruction generates a zero at the end of the first message).

Line 57 does the actual screen printing, through the subroutine COUT, as explained above. **Line 58** increments the Y-Register to point at the next character in the message. This causes **line 55** to load the Accumulator with the next character in the string pointed at by MSGPTR. **Line 59** branches when Y is nonzero, which is always true (as long as the string being printed has fewer than 255 bytes). Finally, **line 60** ends the subroutine and returns program control to the line following the subroutine call.

The actual string data is found in **lines 66-70**. As mentioned above, the first string, MSG1, contains the ASCII codes of the message, followed by a zero byte. The second string, MSG2, begins with two bytes of $8D. This is the ASCII code for a carriage return; therefore, the second string is printed two lines below the first. The second string ends with another carriage return (**line 70**) and the necessary zero byte that indicates the end of the string. Notice the use of the pseudo-opcode ASC, which generates the ASCII code of the string. Most assemblers support this directive; consult your assembler's manual to see if your assembler supports ASC.

AUTOTAB

AUTOTAB is a utility that positions the cursor for PRINT statements in Applesoft programs. Instead of the cumbersome syntax:

```
HTAB 10: VTAB 7
```

for example, you can simply type:

```
&T10,7
```

when AUTOTAB is installed. Moreover, AUTOTAB can automatically center a line of print if you precede the PRINT command with &C40 (in 40-column mode) or &C80 (in 80-column mode).

Key AUTOTAB (**Listing 4**) into your assembler, assemble the program, and save both the source file and the object code (AUTOTAB) to disk.

USING AUTOTAB

To use AUTOTAB, include this line in your Applesoft BASIC program:

```
110 PRINT CHR$(4);"BRUN AUTOTAB"
```

Notice that you must BRUN, not just BLOAD, the program. Once AUTOTAB is installed, use the ampersand command syn-

LISTING 3: MSG.PRINT *(continued)*

```
  44  4019  85 00        STA MSGPTR
  45  401B  A9 40        LDA #MSG2/      ;Do the HOB
  46  401D  85 01        STA MSGPTR+1
  47  401F  20 23 40     JSR MSGPRINT    ;Go print the message
  48  4022  60           RTS             ;Return to BASIC
  49
  50                  ********************************************
  51                  *  MSGPRINT (Message print routine):       *
  52                  ********************************************
  53
  54  4023  A0 00     MSGPRINT LDY #0          ;Zero the index
  55  4025  B1 00     MSGLOOP  LDA (MSGPTR),Y  ;Get a character
  56  4027  F0 06              BEQ ENDMSG      ;If zero, end
  57  4029  20 ED FD           JSR COUT        ;Not zero, so print it
  58  402C  C8                 INY             ;Bump the index
  59  402D  D0 F6              BNE MSGLOOP     ;Always loop
  60  402F  60         ENDMSG  RTS             ;Return to caller
  61
  62                  ********************************************
  63                  *  DATA:                                   *
  64                  ********************************************
  65
  66  4030  CE C9 C2  MSG1     ASC "NIBBLING AT ASSEMBLY HAS IT ALL!"
            C2 CC C9 CE C7 A0
            C1 D4 A0 C1 D3 D3
            C5 CD C2 CC D9 A0
            C8 C1 D3 A0 C9 D4
            A0 C1 CC CC A1
  67  4050  00                 DFC 0
  68  4051  8D 8D     MSG2     DFC $8D,$8D
  69  4053  CD D3 C7           ASC "MSGPRINT PRINTS MESSAGES."
            D0 D2 C9 CE D4 A0
            D0 D2 C9 CE D4 D3
            A0 CD C5 D3 D3 C1
            C7 C5 D3 AE
  70  406C  8D 00              DFC $8D,0

000  Errors

4000  Hex Start of Object
406D  Hex end of Object
006E  Hex Length of Object
7BB0  Hex end of Symbols
```

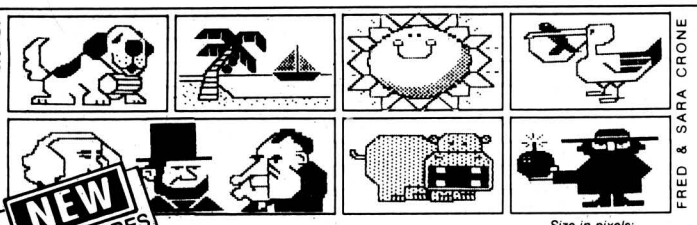
END OF LISTING 3

LISTING 4: AUTOTAB

```
 0                  ;
 1                  ************************************************
 2                  *                                              *
 3                  *                  AUTOTAB                     *
 4                  *                                              *
 5                  *            by S. Scott Zimmerman             *
 6                  *                                              *
 7                  *             Copyright (c) 1986               *
 8                  *             by MicroSPARC, Inc               *
 9                  *             Concord, MA  01742               *
10                  *                                              *
11                  *        The MicroSPARC Assembler 3.0          *
12                  *                                              *
13                  *==============================================*
14                  *                                              *
15                  *  To activate AUTOTAB, BRUN it at the first of *
16                  *  your Applesoft program.  The commands are:  *
17                  *                                              *
18                  *  CURSOR TAB:  &Tn,m  where n is HTAB and m is *
19                  *               VTAB                           *
20                  *                                              *
21                  *  CENTER PRINTING:  &Cn  where n 40 or 80 for *
22                  *            40- or 80-column screen display.  *
23                  *                                              *
24                  ************************************************
25
26                          ORG $0300       ;Page 3 user's space
27
28                  ********************************************
29                  *  EQUATES:                                *
30                  ********************************************
31
32                  ROWNUM   EQU 7          ;Current row number
33                  NCHAR    EQU 9          ;No. characters in PRINT
34                  CH       EQU $24        ;Horiz cursor position
35                  TXTPTR   EQU $B8        ;Points to Applesoft text
36                  AMPER    EQU $3F5       ;Ampersand vector adrs
37                  GETBYTC  EQU $E6F5      ;Skip char, eval exprsn
38                  COMBYTE  EQU $E74C      ;Check comma, eval expr
39                  TABV     EQU $FB5B      ;Monitor TAB routine
40
41                  ********************************************
42                  *  INITIALIZE:                             *
43                  ********************************************
44
45  0300  A9 4C             LDA #$4C        ;JMP opcode
46  0302  8D F5 03          STA AMPER       ;Stuff in amper vector
47  0305  A9 10             LDA #START      ;Load starting address
```

continued on page 106

AD #242



FRED & SARA CRONE

NEW 200 PICTURES ON DISK

MINIPIX™ Disk #1

Size in pixels: 52 high x 88 wide

$29.95 (compatible with any Apple II—DOS 3.3 and ProDOS)

200 PICTURES ON DISK—FOR **SHAPE MECHANIC, PRINT SHOP,™** APPLE MECHANIC, ALPHA PLOT, BEAGLE GRAPHICS, and most other Apple-compatible hi-res graphics programs—

Use pictures "as is", or create instant **MIRROR IMAGES & BLOWUPS.** Plus, convert part of any normal hi-res picture to PRINT SHOP™ format.

PRINT SHOP™ owners: Use Minipix pictures directly on your greeting cards, banners and signs. SHAPE MECHANIC and APPLE MECHANIC fans: Use Minipix in shape table format for hi-res screen displays. ALPHA PLOT and BEAGLE GRAPHICS users: "Clip" Minipix images off the screen and "paste" them onto your own hi-res pictures.

"PRINT SHOP" is a trade mark of Broderbund Software, Inc.

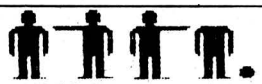
NEW

SHAPE MECHANIC™

by BERT KERSEY & MARK SIMONSEN

$39.95 (any Apple II: DOS 3.3 & ProDOS)

Shape Mechanic is a lightning-fast remake of our *Apple Mechanic* and *Typefaces* disks, combined & rewritten to support ProDOS as well as DOS 3.3.



HI-RES SHAPE MAKER

Shape Mechanic converts your hi-res drawings into "shape tables" so you can create title screens, charts and animation with Apple's built-in *Draw & Xdraw* commands. "List & Learn" demos on the disk teach you how to create your own hi-res programs.

NO FEE to use Mechanic's routines in programs that you sell. Just give us credit.

SHAPE CAPTURE: Convert all or part of any hi-res picture into a shape table.

HI-RES TYPE

30 proportionally spaced hi-res screen display fonts (both large & small) are included on the Shape Mechanic disk. Every character may be edited to become any symbol.

SHADOW OUTLINE CHOMP
MODS COMPUTE SKINNY
FATSO L.E.D. EMBOSSED
SQUARE ZOOLOO

SAMPLE TYPE: Most of these fonts were on our old *"Typefaces"* disk—in DOS 3.3 only.

NEW

FONT MECHANIC™

by MARK & JON SIMONSEN

$29.95 (any Apple II: DOS 3.3 & ProDOS)

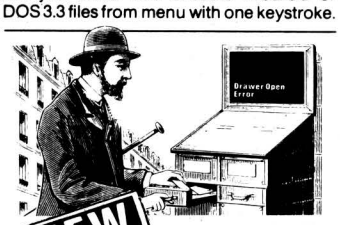
30 all-new hi-res fonts for **SHAPE MECHANIC** and **APPLE MECHANIC!**

SAMPLE FONT MECHANIC TYPE:

Old English CRATE FINE
ZEBRA UNCLE BRANDING
Caesar HOLLYWOOD
BELLS SANDWICH TIMES
U.S. OPEN EDISON
COUNTDOWN ROOSEVELT

FONT UTILITIES: Convert other fonts (DOS Tool Kit™, etc.) into shape fonts. Adjust and distort (stretch, shrink, squish, etc.) for copy-fitting and special effects.

NEW CAT-MENU: Super high-speed; lets you select and execute ProDOS or DOS 3.3 files from menu with one keystroke.



NEW

All Beagle Bros disks are unlocked & copyable.

Don't settle for less.

```
1 HOME: INVERSE: PRINT CHR$(21)
2 X=RND(1)*24+1: VTAB X: HTAB X:
  PRINT X;: HTAB 26-X: PRINT X;: GOTO 2
```

PRO-BYTER™

by BERT KERSEY & JACK CASSIDY

$34.95 (Apple IIe/IIc or 64K II+: ProDOS)

BYTEZAP PRO: All NEW—Inspect ProDOS & 3.3 disks at the byte level. Make normally-impossible changes and repairs.

Find any word in a disk or file. Repair zapped disk bytes and customize disks. Easy instructions show how to do it.

MACHINE-LANGUAGE SORT: One Call in your Applesoft program alphabetizes words—*fast!* Sorts string-arrays—*A$(1)* etc.

MORE: Save time & date in your catalogs. Catalog all directories on ProDOS disks. Swap Applesoft commands, TYPE text files without booting your word processor...

MACHINE LANGUAGE EDITOR: Type in programs from mags, and insert and delete.

ALL-NEW APPLE TIPS! Use memory like an extra disk drive; Disable LIST so it says "File Locked"; Disable/swap ANY command; Rename "Startup"; Personalize catalogs...

NEW

BIG U™

ProDOS™ UTILITY by RANDY BRANDT

$34.95 (Apple IIc, IIe or 64K II+: ProDOS)

FILE MOVER: *(80-columns)* Replaces Apple's *Filer* for ProDOS file transfer (not 3.3). Fast and friendly—no need to re-boot.

CRT WRITER: *(80-col's, IIe or IIc only)* Create fancy 80-column title screens. Or write & print short notes without booting up your word processor. Save screens on disk.

KEY-CAT 80: *(80-col's)* Hit ctrl-Reset to *instantly* see a disk's catalog in 80-columns. One keystroke runs or loads any program.

RAM LOAD: *(IIc or 128K IIe)* Auto-load files to and from RAM. Load all of your favorite programs into memory at once.

18 ALL-NEW ProDOS COMMANDS: *(64K min., any Apple)* **COPY** files from disk-to-disk. **XLIST** programs in better format. **ANYCAT** catalogs any DOS. **MON** monitors ProDOS. **SHOW** loads *and* shows pix. **SEE** lists disk programs without loading.

ERROR EDITOR: Rewrite ProDOS's error messages ("Path Not Found", etc.).

INPUTTER: Accepts commas & colons, rejects control-characters. Esc to abort, etc. You pre-determine maximum string length.

REM-ZAPPER: Work with 2 versions of the same Applesoft program—one with Rem's and one without. Switch instantly. **MUCH MORE** including ProDOS versions of many Beagle UTILITY CITY programs.

TRIPLE-DUMP™

by MARK SIMONSEN & ROB RENSTROM

$39.95 (any Apple II: DOS 3.3 & ProDOS)

PRINT ANY APPLE IMAGE: Use your dot matrix graphics-capable printer to print:

1. Hi-Res and Double Hi-Res graphics
2. Lo-Res and Double Lo-Res graphics
3. 40- and 80-Column text-screen text.

Print all kinds of images with absolutely no programming knowledge. OR call T-Dump routines from your Applesoft programs.

SPECIAL EFFECTS: Crop, rotate, magnify, compress, distort, adjust print density, etc.

GIANT BANNERS too: Print big signs and messages (8½" high letters) on any printer.

D CODE™

by ALAN BIRD

$39.95 (any Apple II: DOS 3.3 & ProDOS)

PROGRAM PACKER: Compress Applesoft programs to increase efficiency. Combine program lines, shorten variables or remove Rem's. Find lines that won't execute.

AUTO-PROOFREADER: Errors are caught as you type, *before* a program is run.

SUPER-TRACE: When a program stops or crashes, type "DUMP" to look at the last 10-10,000 statements executed, in order.

"Live" tracing too, with each executing statement, line no. and selected variable value in a window at the bottom of screen.

BREAKPOINTS: Your program stops when you want. For example, when X gets set to 99, or the 3rd time a Gosub occurs.

LIGHTNING-FAST FIND: Find strings & variables *fast*—search big programs in apx. *2 seconds*—very useful! D Code is fully-transparent to your programs, and compatible with GPLE, Double-Take, etc.

EXTRA K™

by MARK SIMONSEN & ALAN BIRD

$39.95 (IIc or 128K IIe: DOS 3.3 & ProDOS)

EXTRA VARIABLES: Applesoft programs run undisturbed in main memory while variables strings, etc., reside in the other 64K.

EXTRA APPLE: Have two 64K Apples & programs in memory (ProDOS & DOS 3.3 at the same time if you want). Switch at will.

EXTRA-FAST COPIES: "Nibble copy" and *verify* your unprotected disks in 35 seconds!

EXTRA FEATURES: Create combo 3.3/ProDOS disks. MANY MORE 128K utilities.

EXTRA SCREENS: Store all kinds of images in memory. Display any one instantly.

SHOW SEVERAL HI-RES PIX PER SECOND!

GPLE™

by NEIL KONZEN

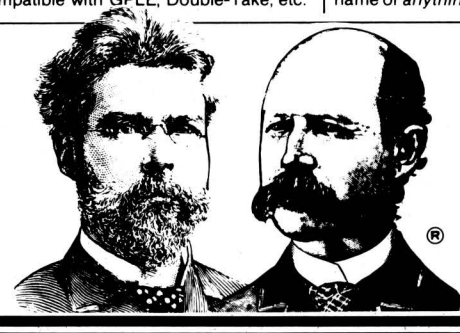
$49.95 (any Apple II: DOS 3.3 & ProDOS)

GLOBAL PROGRAM LINE EDITOR: The best, most popular Applesoft editor. Works like a word processor for Basic program lines.

INSERT AND DELETE: Quickly make changes to programs. No more awkward "escape-editing". No more hassles with extra spaces in quote statements. No more slow "cursor-tracing" to the end of a line.

SEARCH AND REPLACE: Find every occurrence of a string or variable in a program. *Replace* any word or variable too.

FUNCTION KEYS: Define your own—For example, ESC-1 can Catalog drive 1, ESC-N can clear the screen and type your name or *anything*... Customize your Apple!



Phone now for a free Beagle Bros Catalog.

24 Apple II software products under $60.

If you don't have your copy of our Winter/Spring 1986 catalog, phone toll free 1-800-227-3800 ext. 1607.

Beagle Bros MICRO SOFTWARE

3990 OLD TOWN AVE., SAN DIEGO, CA 92110 (PHONE 619-296-6400)

GOTO YOUR SOFTWARE STORE.

Or buy directly from Beagle Bros. WE SHIP *FAST* via First Class Mail.

☐ Alpha Plot	$39.50
☐ Beagle Bag	29.50
☐ Beagle Basic	34.95
☐ Beagle Graphics	59.95
☐ Big U	34.95
☐ D Code	39.95
☐ DiskQuik	29.50
☐ DOS Boss	24.00
☐ Double-Take	34.95
☐ Extra K	39.95
☐ Fatcat	34.95
☐ Flex Type	29.50
☐ Font Mechanic	29.95
☐ Frame-Up	29.50
☐ GPLE	49.95
☐ I. O. Silver	29.95
☐ Minipix Disk #1	29.95
☐ Pro-Byter	34.95
☐ ProntoDOS	29.50
☐ Shape Mechanic	39.95
☐ Silicon Salad	24.95
☐ Tip Disk #1	20.00
☐ Triple-Dump	39.95
☐ Utility City	29.50

Visa/MasterCard or COD, order Toll Free
All 50 states, 24 hrs. every day (orders only)

1-800-227-3800 ext.1607

OR MAIL US A CHECK, MONEY ORDER OR VISA/MASTERCARD NO.

NAME _____

ADDRESS _____

_____

CITY, STATE _____ ZIP _____

VISA MasterCard CARD NO. _____

EXPIRATION DATE _____

Add $1.50 First Class Shipping, any size order
Overseas add $5.50—COD add $3.00—Calif. add 6%

Mail to **BEAGLE BROS, DEPT. A**
3990 OLD TOWN AVENUE, SUITE 102C
SAN DIEGO, CALIFORNIA 92110

ALL ORDERS SHIPPED IMMEDIATELY.

tax described above for tabbing and centering text. For the centering command (&C) to work properly, the PRINT statement following the & command must contain the entire string within quotes (i.e., "STRING") and not in a string variable (such as A$), nor in an Applesoft expression. AUTOTAB only counts the number of characters within the quotes to calculate the centering location. Make sure that the string contains fewer than 40 characters in 40-column mode and fewer than 80 characters in 80-column mode.

AUTOTAB.DEMO (**Listing 5**) is a short Applesoft program that demonstrates the use of AUTOTAB.

HOW AUTOTAB WORKS

Lines 32-39 of AUTOTAB contain the EQUates to Applesoft and the Monitor ROM routines. ROWNUM and NCHAR are variables defined by the programmer, while the other EQUates are ROM locations or variables used by ROM. For example, TXTPTR is probably the most important pointer used by the Applesoft BASIC interpreter. TXTPTR (text pointer) points to the current Applesoft character in memory. AUTOTAB uses TXTPTR in the text centering routine.

AMPER (**line 36**) is the ampersand (&) vector address. (A vector is a sequence of bytes that redirects program control to another region of memory.) **Lines 45-50** set up the & vector by first putting the JMP code ($4C) at AMPER (address $3F5). AUTOTAB then puts the LOB of the program starting address at AMPER+1 ($3F6), and puts the HOB of the program starting address at AMPER+2 ($3F7).

The main program starts in **line 57** at address $0310. This is the address to which the & vector points. When the Applesoft interpreter encounters an & in the program code, it does three things:

1. Increments the text pointer, TXTPTR, to point to the character immediately following the &
2. Loads the Accumulator with the byte value of the character after the &
3. JMPs to AMPER ($3F5)

If a JMP code is located at $3F5 (which is the case after BRUNing AUTOTAB), the program JMPs to the address stored at AMPER+1 and AMPER+2, which, in this case, contains the address $0310.

Lines 57-61 of AUTOTAB evaluate the byte in the Accumulator (the one immediately following the &) to see if there is a T or a C, and if so, it branches to the appropriate section of code.

Lines 67-73 contains the TAB routine. The Applesoft ROM routine GETBYTC is used to evaluate the expression following the T in the & call. The Applesoft expression may contain any combination of variables, constants, and arithmetic or logical opera-

LISTING 4: AUTOTAB *(continued)*

```
   48  0307  8D F6 03           STA AMPER+1      ;Save starting address
   49  030A  A9 03              LDA #START/      ;Now get HOB
   50  030C  8D F7 03           STA AMPER+2      ;And save it, too
   51  030F  60                 RTS
   52
   53                    ;*****************************************
   54                    ;  PROGRAM START:                        *
   55                    ;*****************************************
   56
   57  0310  C9 54      START    CMP #$54         ;Is it 'T'?
   58  0312  F0 05              BEQ TAB          ;Yes, go set tab
   59  0314  C9 43              CMP #$43         ;Was it 'C'?
   60  0316  F0 0F              BEQ CENTER       ;Yes, go center print
   61  0318  60                 RTS              ;None of above, so quit
   62
   63                    ;*****************************************
   64                    ;  TAB the cursor:                       *
   65                    ;*****************************************
   66
   67  0319  20 F5 E6   TAB      JSR GETBYTC      ;Eval expression of HTAB
   68  031C  CA                  DEX              ;Make range 0-39
   69  031D  86 24               STX CH           ;Store in mon HTAB loc
   70  031F  20 4C E7            JSR COMBYTE      ;Eval expression for VTAB
   71  0322  CA                  DEX              ;Make range 0-23
   72  0323  8A                  TXA              ;Put in accum for TABV
   73  0324  4C 5B FB            JMP TABV         ;Tab & return to BASIC
   74
   75                    ;*****************************************
   76                    ;  CENTER printing:                      *
   77                    ;*****************************************
   78
   79  0327  20 F5 E6   CENTER   JSR GETBYTC      ;Evaluate expression for
   80  032A  86 07               STX ROWNUM       ; # of chars/line & save
   81  032C  A9 00               LDA #0           ;Zero the print chars
   82  032E  85 09               STA NCHAR        ;No. characters in string
   83  0330  A0 01               LDY #1           ;Init the search index
   84
   85  0332  B1 B8      PRINTST  LDA (TXTPTR),Y   ;Get byte value from prgm
   86  0334  C8                  INY              ;Increment to next char
   87  0335  C9 22               CMP #$22         ;Is it a quote mark?
   88  0337  D0 F9               BNE PRINTST      ;No, go get next char
   89
   90  0339  B1 B8      STRCOUNT LDA (TXTPTR),Y   ;Yes, so count chars
   91  033B  C8                  INY              ;Go to next character
   92  033C  C9 22               CMP #$22         ;Is it a quote?
   93  033E  F0 09               BEQ CALTAB       ;Yes, end of string
   94  0340  C9 20               CMP #$20         ;Is it a control char?
   95  0342  90 02               BCC CTRL         ;Yes, so don't inc count
   96  0344  E6 09               INC NCHAR        ;Increase character count
   97  0346  4C 39 03   CTRL     JMP STRCOUNT     ;Go check next character
   98
   99                    ;  Calculate the proper TAB for centering.
  100
  101  0349  38         CALTAB   SEC              ;Calculate diff between
  102  034A  A5 07               LDA ROWNUM       ; no. characters per row
  103  034C  E5 09               SBC NCHAR        ; and no. chars in string
  104  034E  4A                  LSR              ;Divide the result by two
  105  034F  85 24               STA CH           ;Stuff into HTAB value
  106  0351  60                 RTS              ;Return to BASIC

000  Errors

0300  Hex Start of Object
0351  Hex end of Object
0052  Hex Length of Object
7B81  Hex end of Symbols
```

END OF LISTING 4

LISTING 5: AUTOTAB.DEMO

```
10  REM ************************
20  REM *     AUTOTAB.DEMO     *
30  REM * BY SCOTT ZIMMERMAN *
40  REM * COPYRIGHT (C) 1986 *
50  REM * By MICROSPARC, Inc *
60  REM * CONCORD, MA  01742 *
70  REM ************************
80  REM
90  TEXT : HOME : IF  PEEK (1014) = 16 AND  PEEK
    (1015) = 3 THEN 110
100  PRINT  CHR$ (4);"BRUN AUTOTAB"
110  & C40: INVERSE : PRINT " AUTOTAB ": NORMAL
    : & T1,3: & C40: PRINT "BY S. SCOTT ZIMM
    ERMAN": & C40: PRINT "COPYRIGHT (C) 1986
     BY MICROSPARC"
120  & T3,10: PRINT "->AUTOTAB AUTOMATICALLY
    TABS."
130  & T1,13: & C40: PRINT "->IT AUTOMATICALL
    Y CENTERS.<-"
140  & T3,19: PRINT "JUST REMEMBER TO BRUN AU
    TOTAB AND": PRINT "THEN USE &-COMMANDS A
    S SHOWN IN THIS": PRINT "DEMONSTRATION P
    ROGRAM."
```

END OF LISTING 5

tions. If the program contains the command &T10,5, GETBYTC would cause TXTPTR to skip the T, evaluate the expression, and put the result (10 or $0A) into the X-Register. Since Applesoft tabs use the range 1-40, but CH and TABV (see above) use the range 0-39, AUTOTAB decrements the X-Register in **line 68** before storing the results in CH.

After GETBYTC evaluates an expression, TXTPTR is left pointing to the character following the expression. COMBYTE is a related Applesoft ROM routine that checks for a comma before evaluating an expression. This is just what we want for evaluating the expression after the comma in &T10,5. Hence, **line 70** calls COMBYTE which, in this example, puts a five in the X-Register. AUTOTAB decrements the X-Register (**line 71**) to adjust the tab range, transfers the result from the X-Register to the Accumulator (TABV requires the vertical tab in the Accumulator), and calls the TABV routine. Notice that AUTOTAB could have used a JSR TABV followed by an RTS to end the routine, but instead it simply JMPs to TABV and thereby saves one byte.

The CENTER routine is a little more complicated. It uses GETBYTC (**line 79**) to check for 40-column or 80-column centering, and stores the result in the variable ROWNUM (**line 32**). **Lines 81-82** zero the NCHAR, used to count the number of characters in the PRINT string.

At this point in the program, the Applesoft text pointer, TXTPTR, is pointing at the character in the BASIC line of code following the &C40 or &C80 command. AUTOTAB now uses TXTPTR to count the number of characters in the next PRINT string. This starts on **line 85**, which is the beginning of a loop to search for the first quote mark (") in the print string. The command:

LDA (TXTPTR),Y

which is the indirect indexed addressing mode, loads the byte value of a character in the BASIC line, INY increments the index to point to the next character, and CMP #$22 checks whether the number in the Accumulator is the ASCII code for a quote mark.

When AUTOTAB finds the first quote mark, the program drops down to **line 90**, where it begins a search for the second quote mark, counting the number of characters in between.

Once AUTOTAB finds the second quote mark, the program goes to **line 101**, where it calculates the HTAB necessary to center the string. I will not try to explain the calculations at this time.

In the next installment of Nibbling At Assembly Language we'll discuss assembly language arithmetic. In the meantime, analyzing AUTOTAB will challenge your assembly language programming skills.

. . . the object code is only 273 bytes, compared to 347 bytes for the BASIC program.

Dr. Zimmerman has just co-authored (with Beverly Zimmerman) a book entitled Action Games for the Apple: How to Design Computer Games, *(Scott, Foresman and Co., 1900 East Lake Ave., Glenview, IL 60025). The book is designed for Applesoft programmers, and contains software tools, tips and techniques for programming sound effects, music, graphics and animation. The book includes three utilities (with assembly language subroutines) and 11 example games.*

SCRG SOUTHERN CALIFORNIA RESEARCH GROUP

INSTANT
APPLEWORKS™

Now, you can have **APPLEWORKS**™ *built-in* your APPLE //e™. With our programming service, your copy of **APPLEWORKS,** and the **quik-Loader**™, your computer can be ready within 2 seconds after applying power.

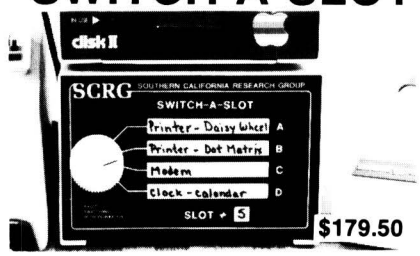
After receiving your order, we will immediately send you a kit containing everything you'll need. (Units purchased through dealers will contain the kit). Copy your original **APPLEWORKS** program onto the disks included in the kit. Configure the program the way you want it used, and return it to us in the envelope provided. We will program the necessary "chips", install them in the **quikLoader**, verify proper operation, and rush it to you. When you receive it, just plug it into *any* peripheral slot. No more searching for your disks, waiting for the startup disk to load, swapping to the program disk, and waiting even longer. The quikLoader is faster and more reliable than even a hard disk drive. Updates should be no problem, as the "chips" are re-programmable.

quikLoader– $179.50
programming service– 89.50
Total– $269.00

APPLEWORKS and APPLE //e are trademarks of APPLE COMPUTER INC.
quikLoader (designed by Jim Sather) is a trademark of S.C.R.G.

SEE A DEMO AT YOUR DEALER

SWITCH-A-SLOT



$179.50

Designed by Joy George

The **SWITCH-A-SLOT** is an expansion chassis, which allows the user to plug in up to four peripheral cards in one slot. One of these cards is selected for use, and *only* that card draws power.

This product is especially useful where the software requires the printer to be in a particular slot, and the user wishes to choose between two or more printers.

SWITCH-A-SLOT and **EXTEND-A-SLOT** work well with most slow to medium speed cards, such as modems, printers, clock, music, etc. They are not recommended for high speed data transfer devices such as disk drive controllers, alternate processor, and memory cards.
Due to the nature of these products, and unit-to-unit variations of computers and peripherals, we are unable to guarantee that these products will work with any particular system. However, as with all our products, we offer a ten-day trial period. If our product does not operate to your satisfaction, return it for a prompt refund.

EXTEND-A-SLOT

The **EXTEND-A-SLOT** brings a slot outside your APPLE™, allowing an easy change of cards. The 18" (45cm) cable is long enough to allow placement of the card in a convenient location. the high quality connectors are gold plated for reliability. **$34.95**

SLOT 3 CLOCK™

Designed by Chuck Shaffer

The **SLOT 3 CLOCK** will plug into any slot of the APPLE][,][+, or //e. The main feature is that, unlike most cards, it can plug into slot 3 of the //e *without interfering* with the operation of the Extended 80-column card. **PRODOS**™ is fully supported, and the card is APPLESOFT™ compatible. Installation software and a long-life DURACELL™ lithium battery are included.

$79.50

PROmGRAMER™

Hardware design by Bob Brice — Software by Bob Sander-Cederlof

The PROmGRAMER is an inexpensive **EPROM** (**E**rasable **P**rogrammable **R**ead **O**nly **M**emory) programmer for the **APPLE**][,][+, and //e computers. The unit plugs into any slot of the computer, and allows the user to program any standard 5 volt, 27 series **EPROM**. Although not intended as a production tool, the ease of use allows rapid programming, copying, duplication, or modification of **EPROM**s.

$179.50

Paddle-Adapple

The **PADDLE-ADAPPLE** is a game plug expansion adapter that plugs into the interior game I/O socket, and is designed to operate in one of two modes. In the first, it allows you select between one of two devices, such as Koala Pad™ and joystick. The device is selected by the flip of a switch. In the second mode, the **PADDLE-ADAPPLE**, with appropriate software, allows the use of two joysticks simultaneously to allow use with games such as **ONE-ON-ONE**™ and **ARCHON**™.

There are three versions to adapt to any combination of the newer type APPLE connector (the 9-pin 'D' sub-miniature), or the older 16-pin plug.

- The **PADDLE-ADAPPLE** has two 16-pin sockets.
- The **PADDLE-ADAPPLE COMBO** has one 'D' subminiature and one 16-pin connector.
- The **PADDLE-ADAPPLE 'D'** has two subminiature 'D' (9-pin) connectors.

$29.95

9-16 Adapter
for APPLE //e & //c

The 9-16 adapter permits the use of most 16-pin I/O devices with the APPLE //e or //c. Plugging this adapter into the subminiature 'D' connector allows the use of 16-pin device, such as the **PADDLE-ADAPPLE**, paddles, joystick, 16 pin Koala Pad™, etc. The only limitations are those devices that use the annunciators or the $C0$40 strobe. **NOTE**–the //c does not support two joysticks. **$14.95**

16-9 Adapter
for APPLE][and][+

With this adapter, owners of early APPLEs can take advantage of the newer 9-pin game products, such as paddles, joysticks, **MUPPET LEARNING KEYS**™, etc. **NOTE** – If you have more than one game I/O device, consider purchasing our **PADDLE-ADAPPLE COMBO** or **PADDLE-ADAPPLE 'D'** instead. **$14.95**

at your local dealer or direct from:
SOUTHERN CALIFORNIA RESEARCH GROUP
Post Office Box 593-N
Moorpark, CA 93020
Telephone (805) 529-2082
Telex: 658340 - SCRG (Intertel SNC)
Shipping charges: **U.S. & Canada** $2.50
Other: Switch-A-Slot; $10.00 surface; $25.00 air. Other products $5.00 surface; $10.00 air.

SIX MONTH WARRANTY
TEN DAY RETURN PRIVILEGE
TOLL-FREE ORDER LINES
NAT'L (800) 635-8310
CALIF. (800) 821-0774
Information (805) 529-2082
VISA & MASTERCARD ACCEPTED

ProDOS DIRECTORY LIST

by Keith Stattenfield

TIPS 'N TECHNIQUES

Use this ProDOS

utility to list all the files on a disk, including all subdirectories, to a 40- or 80-column screen or a printer.

ProDOS is a big step forward in both convenience and organization. Unfortunately, to anyone familiar with DOS 3.3, some things seem very difficult, while others are downright impossible. One difficulty is finding out exactly what is on a disk. ProDOS allows for multiple catalogs. When you type CATALOG, a directory appears that may contain several subdirectory files. Typing CATALOG followed by a space and the name of one of these subdirectories displays a new catalog. This system creates a capacity for file organization: one subdirectory may contain utility programs, while another contains data files, and yet another contains text files. ProDOS allows for the interface of hard disks, and on a hard disk, this type of file organization is far superior to that provided by DOS 3.3. However, on floppy disks, this system can be somewhat confusing.

It creates a problem. To see all the files on a disk, you have to catalog each subdirectory separately, a task that is time-consuming and error prone. ProDOS Directory List rescues you from this task. This utility program lists all the files in each of the directories on any ProDOS disk.

USING THE PROGRAM

To use the program, simply RUN it and enter the path name of the base directory (or the slot and drive of the disk that contains the base directory). The base directory is the directory in which the program starts looking. If it finds any subdirectories it will list them, and then any subdirectories in these subdirectories are listed, and so on.

If you specify the volume directory for a disk, all of the files on the disk will be listed. If you specify the slot and drive of the disk, the program will find the name of the volume directory of the disk in that drive and use that name as the base name.

After the path name has been entered and checked, the program asks whether or not to send the output to the printer. Currently, the program assumes that the printer is in slot 1, and the format string to be sent to the printer is in **line 130**. (If your setup is different, see HOW IT WORKS below.) If you want to send the listing to the printer, press Y; if not, press N. If you press N, then the program will ask whether to send output to an 80-column card. If you don't have an 80-column card or if you only want to see 40 columns of the catalog, press N. If you press Y, the program expects the 80-column card in slot 3.

Next, the listing is displayed. Press < CTRL > S to pause if you are listing to the screen. After the listing finishes, you have the option to display it again.

ENTERING THE PROGRAM

To key in the program, type in the Applesoft program shown in **Listing 1** and save it with the command:

SAVE DIRECTORY.LIST

For help in entering *Nibble* listings, see "A Welcome to New *Nibble* Readers" at the beginning of this issue.

HOW IT WORKS

Unlike DOS 3.3, ProDOS allows an OPEN operation on any type of file. This means that if you open a directory file, ProDOS will return the catalog line by line. ProDOS Directory List takes the following steps to list the catalog:

1. Opens the base directory as a file of type DIR.
2. Reads the three descriptor lines at the beginning of each catalog.

Keith Stattenfield, 908 Perry Ave., Racine, WI 53406. ProDOS Directory List is compatible with ProDOS only.

3. Reads a line of the catalog.
4. Checks the MID$ of each catalog entry from the 18th character to the 20th character. This is the file type. If this string is DIR, then the complete path name of the current file is saved in the end position of a string array.
5. Goes back to step 3 until the last line of the current catalog has been read.
6. Checks whether the end of the array, which contains the names of the next subdirectory, has been reached. If not, the file named in the next element in the array is opened as a file of type DIR and program flow returns to step 2.

This utility program lists all the files in each of the directories on any ProDOS disk.

The program assumes that the printer is in slot 1. If your printer is in a different slot, change the number assigned to the variable PR% in **line 170** (**Listing 1**) accordingly. Also, change the format string in **line 130** to conform to your printer.

One interesting part of the program is the "input anything" routine used to get the path name. Its use is simple. The statement:

CALL 768,A$:A$=A$+" "

returns the string in A$.

APPLICATIONS

ProDOS Directory List could be inserted into another program to streamline the process of finding a file in a subdirectory of several disks. It could also be converted to machine language and a new command added to ProDOS (perhaps something like FIND *filename*), which would scan all of the current devices and return the path name of the file in question.

This same technique could be used to list only certain file types in a catalog, or only locked files, for instance. Since the file type and file status can be easily determined from the CATALOG string, listing just the files with specific characteristics would be very easy.

Nibble Light Pen, ProDOS Directory List, ProCursor, DISPLAY and programs from Nibbling at Assembly Language V are available on diskette for an introductory price of $17.95 plus $1.50 shipping/handling ($2.50 outside the U.S.) from Nibble, 45 Winthrop St., Concord, MA 01742. Introductory price expires 3/31/86.

LISTING 1: DIRECTORY.LIST

```
10  REM ************************
20  REM *    DIRECTORY.LIST     *
30  REM * BY KEITH STATTENFIELD *
40  REM *  COPYRIGHT (C) 1986   *
50  REM *  BY MICROSPARC, INC   *
60  REM *  CONCORD, MA  01742   *
70  REM ************************
```

```
80  REM
90  REM  FS$ CONTAINS THE PRINTER SETUP STRIN
    G.
100 REM  IT IS CURRENTLY SET UP FOR THE GRAP
    PLER INTERFACE
110 REM   AND APPLE DMP PRINTER.
120 REM
130 FS$ =  CHR$ (9) + "80N" +  CHR$ (9) + "60
    P" +  CHR$ (27) + "E" +  CHR$ (27) + "L0
    10"
140 REM
150 REM  PR% IS THE PRINTER SLOT
160 REM
170 PR% = 1
180  GOTO 280
190 REM
200 HTAB 20 -  LEN (A$) / 2 + 1: PRINT A$: RETURN

210  INVERSE : GOSUB 200: NORMAL : RETURN
220 GOSUB 200:A$ =  LEFT$ ("----------------
    ------------------------", LEN (A$)): GOSUB
    200: RETURN : REM 40 -'S
230 VTAB 9: HTAB 1: CALL  - 958: PRINT : RETURN

240 A$ = "PRESS <RETURN> TO CONTINUE": VTAB 2
    3: GOSUB 200: POKE  - 16368,0: WAIT  - 1
    6384,128
250 VTAB 23: CALL  - 868: POKE  - 16368,0: RETURN

260 GET A$:YES = (A$ = "Y") + (A$ = "Y"): ON
     NOT (YES + (A$ = "N") + (A$ = "N")) GOTO
    260: PRINT A$: RETURN
270 REM
280 REM  -- START OF MAIN --
290 REM  --    PROGRAM     --
300 TEXT : HOME : DIM DIR$(100)
310 PRINT  CHR$ (17);
320 VTAB 2:A$ = "DIRECTORY.LIST": GOSUB 220
330 PRINT :A$ = "  BY KEITH STATTENFIELD": GOSUB
    200: PRINT :A$ = "* COPYRIGHT (C) 1986 B
    Y MICROSPARC,INC *": GOSUB 200
340 VTAB 8: PRINT "-------------------------
    ---------------": REM 40 -'S
350 PTR = 768:DL$ = "-------------------------
    ----------------------------------------
    ----------------": REM !80 DASHES !
360 READ A: IF A >  - 1 THEN  POKE PTR,A:PTR
     = PTR + 1: GOTO 360
370 POKE 34,8: HOME
380 D$ =  CHR$ (4)
390 PRINT D$"PREFIX": INPUT OP$
400 GOSUB 230: PRINT "ENTER THE NAME OF THE
    VOLUME OR": PRINT "DIRECTORY FOR THE BAS
    E TO LIST"
410 PRINT : PRINT "DEFAULT IS ";OP$
420 PRINT : PRINT "SLOT AND DRIVE CAN BE SPE
    CIFIED (,S#,D#)"
430 VTAB 18: PRINT ":";OP$
440 VTAB 18: HTAB 2: CALL 768,NP$:NP$ = NP$ +
    ""
450 IF NP$ = "" THEN NP$ = OP$
460 VTAB 18: HTAB 2: PRINT NP$
470 ONERR  GOTO 490
480 PRINT : PRINT D$"PREFIX"NP$: PRINT D$"PR
    EFIX": INPUT NP$: GOTO 500
490 GOSUB 230:A$ = "ERROR #" +  STR$ ( PEEK
    (222)) + " IN PREFIX": GOSUB 210: GOSUB
    240: RUN
500 POKE 216,0: VTAB 18: HTAB 2: PRINT NP$;:
     CALL  - 958: PRINT
510 VTAB 23: PRINT "LIST CATALOG ON PRINTER?
    ";: GOSUB 260
520 PR = 0: IF YES THEN PR = PR%
530 VTAB 23: HTAB 1: CALL  - 868: IF  NOT PR
     THEN  PRINT "LIST ON 80-COLUMN SCREEN?"
    ;: GOSUB 260: IF YES THEN PR = 3
540 IF PR = PR% THEN  PRINT FS$
550 LN = 39 + (40 * (PR > 0))
```

continued on next page

LISTING 1: DIRECTORY.LIST *(continued)*

```
560 P1 = 1:P2 = 1:DIR$(1) = NP$
570  REM
580  REM  - PERFORM THE CATALOG -
590  REM
600  TEXT : HOME
610  PRINT : PRINT D$"PR#";PR: IF PR = PR% THEN
      PRINT FS$
620  PRINT D$"OPEN "DIR$(P1)",TDIR"
630  PRINT D$"READ "DIR$(P1)
640  INPUT T$: PRINT "CATALOG OF ";DIR$(P1)
650  INPUT T$: PRINT  LEFT$ (T$,LN)
660  INPUT T$: PRINT  LEFT$ (DL$,LN)
670  ONERR  GOTO 770
680  INPUT NAME$:NAME$ = NAME$ + ""
690  IF NAME$ = "" THEN 770
700 TY$ =  MID$ (NAME$,18,3)
710  PRINT  LEFT$ (NAME$,LN)
720  IF TY$ <  > "DIR" THEN 680
730 NAME$ =  MID$ (NAME$,2,16): IF  RIGHT$ (D
    IR$(P1),1) <  > "/" THEN NAME$ = "/" + N
    AME$
740  IF  RIGHT$ (NAME$,1) = " " THEN NAME$ =
     LEFT$ (NAME$, LEN (NAME$) - 1): GOTO 74
    0
750 P2 = P2 + 1:DIR$(P2) = DIR$(P1) + NAME$
760  GOTO 680
770  REM
780  REM  - WE'RE DONE WITH THAT CATALOG,
790  REM     SO CLOSE THE DIRECTORY & DO
800  REM     THE NEXT ONE -
810  REM
820  PRINT : PRINT D$"CLOSE"
830  IF P1 =  < P2 THEN P1 = P1 + 1: GOTO 620
840  IF PR AND PR <  > 3 THEN  PRINT D$"PR#0"
850  VTAB 24: HTAB 1: CALL  - 868: GOSUB 240
860  VTAB 23: HTAB 1: CALL  - 868
870  PRINT "PERFORM ANOTHER CATALOG?";: GOSUB
    260
880  IF YES THEN  RUN
890  TEXT : HOME : END
900  REM
910  REM  - INPUT ANYTHING MACHINE CODE DATA
    -
920  REM
930  DATA 32,190,222,32,227,223,162,0,32,117,
    253,160,0,138,145,131,200,169,0,145,131,
    200,169,2,145,131,32,57,213,96
940  DATA  -1
```

END OF LISTING 1

KEY PERFECT 5.0
RUN ON
DIRECTORY.LIST

CODE-5.0	LINE# - LINE#	CODE-4.0
7C62FD67	10 - 100	7500
DAF88DD9	110 - 200	4291
A690B7C4	210 - 300	A076
1A3753F4	310 - 400	C996
6171C152	410 - 500	795A
4F012FD2	510 - 600	679C
39F36578	610 - 700	6740
BCFC0CCE	710 - 800	7266
F09FFC02	810 - 900	48A9
0D5D3B28	910 - 940	4B3B
8AE7AAF0	= PROGRAM TOTAL =	09B6

APPLEWORKS ENHANCEMENTS

Read why Applied Engineering
is the acknowledged leader in AppleWorks enhancements

- Applied Engineering was first to expand Appleworks beyond 55K.
- Applied Engineering was first to eliminate disk access on Drive 1 in Appleworks, thereby dramatically speeding up the program and reducing disk drive wear.
- Applied Engineering was first to expand AppleWorks to 1 megabyte of memory.
- Applied Engineering was first to expand AppleWorks to 3 megabytes.
- Applied Engineering was first to provide auto segmenting on large files so any size file can be saved on regular floppies.
- Applied Engineering was first to increase the maximum number of records in the data base from 1350 to 16,300.
- Applied Engineering was first to increase the number of lines in the AppleWorks word processor from 2250 to 16,300.
- Applied Engineering was first to display the time and date on the AppleWorks screen.
- Applied Engineering was first to provide auto dating as part of the data base field.
- Applied Engineering was first to provide auto time stamping as part of the data base field.
- Applied Engineering was first to include a built-in print spooler for AppleWorks. So you're back in AppleWorks in 10 seconds not 10 minutes.

With all of these firsts, it's no wonder most people choose Applied Engineering when they want enhancements to Appleworks. Especially when you consider that nearly all the "firsts" listed above are "onlys" too. And our Appleworks enhancements are as easy to use as inserting a disk and pressing return because all enhancements are merged with the AppleWorks program disk and stay there out of sight until needed, only now AppleWorks will run up to 20 times faster, and be easier to use at the same time.

And we're working on new improvements to AppleWorks all the time, so the few remaining criticisms the software reviewers gave AppleWorks will vanish forever. So if you want more out of AppleWorks, order RamWorks II™ and Timemaster H.O.™ for the IIe or Z-RAM™ and IIc System Clock™ for the IIc. You'll find specifications and prices for these products listed in this magazine. Please look for Applied Engineering in the advertiser's index.

There's no question about how they'll increase your productivity. The only question you'll have is how you ever got along without them.

214-241-6060

APPLIED ENGINEERING
"We Set the Standard"

What you should know about the International Apple Core.

If you're like most Apple users, you don't realize the many benefits of an IAC membership. Or what it can do for you and your computer.

So what's an IAC?

International Apple Core™ is a non-profit organization of Apple users and user groups. We are dedicated to providing education, information and support to users of Apple and Apple-compatible products.

We aren't just for clubs. An individual can enjoy all the special benefits we have been providing user groups since 1979. You may share information on new applications. Or learn the latest on products. Or keep up on Apple events. Even receive specialized training.

If you'd like to join a local user group or contact other Apple enthusiasts, we can help you find them.

A membership that pays you

As a member of IAC you nearly get paid with what you will save. By joining you will save nearly $20 on our electronic spreadsheet, *IACcalc*™. Not to mention all the other benefits only available to you as a member.

Information, please.

IAC provides many sources of information. Your $30 yearly membership fee brings a 12-issue subscription to *Nibble* magazine devoted to Apple systems and compatibles. *Nibble* features more than $50 worth of ready-to-run Apple programs in each issue. The programs focus on home, business, education and entertainment. *Nibble* also features new products, reviews, tips and techniques for learning more about your Apple and having fun doing it!

You will be eligible to receive the IAC *Disk of the Month*, a subscription to user contributed software. Each disk presents themes like games, education, or a potpourri.

Get an education.

You'll receive many educational benefits from our software programs. Our 3Pak™ series of theme packaged programs includes Educational, Financial, Games and the popular Logo. We have an Applesoft Tutorial and Education disk which is great for children.

Support for your Apple and you.

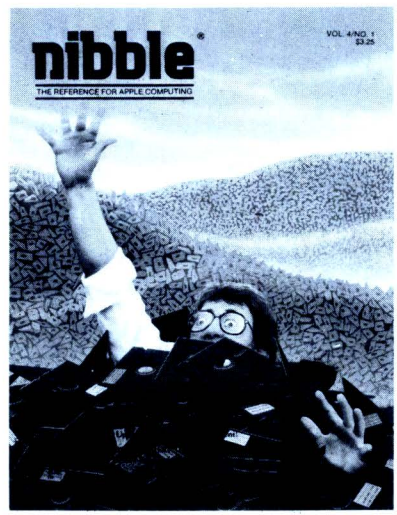
For as little as $50 annually and no deductible, you can protect your hardware, software and disks for up to $15,000.

Attach-Bios and *CP/M→SOS Converter* are ideal disks for Apple II Pascal and Apple III CP/M® application writers, vendors and users. All documentation is on the disks and no softcard is required to run Converter.

Invite your friends and save.

IAC offers user groups a special package. For $90 your group receives a full year IAC membership and a subscription to Nibble. Included are the Disks of the Month and a listing of other IAC affiliated user groups world wide.

Individually or as a group you can't lose. So fill out the membership coupon below. And don't forget to ask for our product catalog.



The companies we keep.

Here is a list of the corporations that support and sponsor the International Apple Core.

Apple Computer Inc., Cupertino, CA
Brock Software Products, Crystal Lake, IL
Burtronix, Orange, CA
Digital Research, Pacific Grove, CA
East Side Software Co., New York, NY
Educational Software Products, Glencoe, IL
Electronic Arts, San Mateo, CA
General Microcomputer, Inc., Michigan City, IN
Hayes Microcomputer, Norcross, GA
Houston Instrument Division, Austin, TX
Howard D Sams Co. Inc., Indianapolis, IN
M&R Enterprises, Santa Clara, CA
Macworld, San Francisco, CA
Multi-Tech Systems, Inc., New Brighton, MN
Nestar Systems, Inc., Palo Alto, CA
Nibble/microSPARC, Inc., Concord, MA
Source Telecomputing, McLean, VA
Sundex Software Corporation, Boulder, CO
Turning Point Software, Watertown, MA
Vagabondo Enterprises, Aptos, CA
Verbatim Corporation, Sunnyvale, CA

Membership Application

☐ $30 U.S. Individual
☐ $40 Canada & Mexico Individual
☐ $72 International Individual
☐ $90 U.S. User Group Package
☐ $152 International User Group Package
☐ I'm already a Nibble subscriber. Please extend my subscription.

Name _____

Address _____

City _____

State/Zip _____ Country _____

All Memberships Include a 12-Month Subscription to Nibble.

Please allow 8-10 weeks for delivery of your first Nibble.

☐ Payment enclosed (U.S. funds only)
☐ VISA ☐ MasterCard
☐ Please send your product catalog.

Charge Card No.

_____

Interbank No. _____ Expiration Date MO DAY YEAR

Signature

INTERNATIONAL APPLE CORE™
Membership Dept. #N-01
908 George Street
Santa Clara, CA 95054
(408) 727-7713

CIRCLE NUMBER 52

ProDOS OUTPUT PROCESSING

by Sandy Mossberg

DISASSEMBLY LINES

The ProDOS BASIC interpreter's output handling routines, along with the relevant Applesoft output routines, are analyzed. In addition, a fix for a BI bug is provided.

The last installment of Disassembly Lines (Vol. 6/No. 11) examined input processing by the ProDOS BASIC interpreter (BI). This time we take on the complex and somewhat top-heavy code associated with character output. The bloated output code stems from the incorporation of trace mode into the BI (DOS 3.3 ignores tracing).

More exciting even than the disassembly was the detective work required to isolate a BI bug that flips trace mode on and off at inopportune moments. If you are a fan of A. Conan Doyle or Agatha Christie, you will find this section downright exhilarating.

OUTPUT STATES

You may want to review the descriptions of I/O hooks, vectors and states in the last D/L column. State numbers designate an index into the I/O state table ($B7F7 for BI 1.1; $B851 for BI 1.0). Whereas input is processed by two state handlers, immediate (KSTATE0) and deferred (KSTATE48), output requires four separate routines: STATE 0 (immediate mode), STATE 4 (deferred mode), STATE 8 (assembling command line in deferred mode), and STATE D (checking for <CTRL>D).

by Sandy Mossberg, 50 Talcott Rd., Rye Brook, NY 10573

APPLESOFT ROM

Before tackling BI output processing (**Listing 1**), we must understand the interaction between Applesoft ROM and the BI. **Listing 2** includes several Applesoft routines, two of which were published in earlier D/L columns (NEWSTT from Vol. 4/No. 2, and OUTDO from Vol. 4/No. 4). Both of the original columns are contained in *Disassembly Lines, Volume 1*, a compendium of articles about Applesoft ROM. Only pertinent sections of the code will be covered here.

First, let us briefly review how Applesoft stores programs. Keywords (i.e., statements or commands) are converted by PARSE ($D559) into tokens, each of which is stored in program memory as a negative ASCII (high bit set) byte. Numerals and variables are kept in positive ASCII (high bit clear) format. When the Applesoft interpreter searches a program and finds a negative ASCII character, a token is inferred and program flow is transferred to the handler for that particular token. We shall observe how this works later on.

All Applesoft statements, immediate and deferred, are processed by NEWSTT (**lines 46-109** of **Listing 2**). After some housekeeping chores, execution begins at **line 75**, where the Applesoft trace flag is tested. If clear (**line 76**), flow drops to GETSTT (**line 86**), where the next program byte is obtained and processed (**lines 87 and 92-103**). If the trace flag is set, **lines 77-78** ensure that immediate mode is not active, and **line 81** prints the trace character (#) by calling OUTDO (**lines 115-125**), the standard Applesoft output subroutine. To make certain that the BI would control all tracing activity, the Apple's software writers resolved that the Applesoft trace flag must always be set so that the BI could gain control when OUTDO calls COUT. (Remember, COUT jumps to the address at $36-$37, which is the current BI intercept.)

Thus, the master plan is to trap all Applesoft new statement processing at OUTRACE (**line 81**) and control true tracing within the BI. As we dissect the BI output code in **Listing 1**, we shall learn how this process works.

IMMEDIATE MODE

The STATE 0 output handler begins by checking if the character to be output is the trace character whose image is stored at $9F61 ($9F98 in BI 1.0) in **line 521** of **Listing 1**. Except for one circumstance, noted later, TRACECHR is the expected pound sign (#). If the trace character is not being printed, flow drops to **lines 109-116**, where the character is echoed if it does not represent a consecutive carriage return (CR). **Lines 120-126** suppress a consecutive CR only if it occurs at the start of a line and a BI command is not pending. (The necessity for this manipulation is beyond my ken. If

someone can enlighten me, I shall be in your debt.) The "command pending" decision is made by testing the first character in the BI text buffer, TXBUF (also called path name buffer and command buffer), for ASCII $3F (the question mark) or greater. Since commands and path names begin with alphabetic characters, it would seem more appropriate to test for ASCII $41 (A) or greater.

If the trace character is being output, X is saved (**line 90**) and **lines 91-96** determine whether the output request comes from OUTRACE (that important **line 81** in NEWSTT). This technique requires explanation. When JSR OUTDO is executed at $D810 (OUTRACE) and JSR COUT is subsequently executed at $DB5F (PRCHR), the stack contains both return addresses (minus one) (see **Table 1**). If the lower address is $D812, OUTRACE output is confirmed, in which case, control drops to **lines 101-105**, where deferred mode is set and the trace character is printed via the STATE 4 output handler. I suspect that these latter lines were designed to trap inadvertent entry into immediate mode from deferred mode, but I do not understand how OUTRACE can be called from immediate mode because EXECUTE (**lines 75-88** of **Listing 2**) skips OUTRACE if immediate mode is operational. Again, I solicit your help in understanding how **lines 101-105** can ever be executed.

If the trace character is being printed, but Applesoft is not tracing (i.e., output does not come from OUTRACE), **line 97** branches to **lines 130-131**, where X and the trace character are restored and flow drops to the character output routine.

CHARACTER OUTPUT

After placing the true I/O handler addresses at the I/O hooks (**line 137**), the target character is printed. If it is not a CR (**lines 142-143**), exit occurs by restoring the true I/O handlers to the global page and resetting the BI intercepts.

The BI assiduously maintains the Applesoft trace flag at a value of $A5. When the TRACE or NOTRACE command is given in immediate mode, Applesoft kicks flow to the appropriate statement handler (**lines 156-163 of Listing 2**) where either a set carry (TRACE) or clear carry (NOTRACE) is rotated into bit 7 of TRCFLG. (The old skip-a-byte trick is used in **line 157** to create the expression BCC $F288 for the purpose of bypassing the CLC instruction when entry is at TRACE.) When the BI next gains control to output a CR, **lines 147-148** of **Listing 1** test for a change in the Applesoft trace flag. If not found, the BI knows that either NOTRACE or TRACE has been executed in immediate mode. In such a circumstance, TRCFLG would contain either $52 (NOTRACE rotates $A5 right and clears bit 7) or $D2 (TRACE rotates $A5 right and sets bit 7). Therefore, if the trace flag has been changed, **line 150** stuffs the rotated value (e.g., $52 or $D2) into the true BI trace flag (DTRACE), and **lines 151-152** reset TRCFLG to $A5. The CR is restored to the Accumulator (**line 153**) and exit occurs.

DEFERRED MODE

STATE 4

VSTATE4 is the main code for processing output in a running program. The initial segment (**lines 279-287**) is identical to YSTATE0. If output is coming from OUTRACE, control branches to BI trace processing, which is detailed in the next section. If nontrace output is encountered, the entry A and X Registers are restored (**lines 292-293**) and file output activity is tested. If a character is being written to a file and that character is neither the immediate mode prompt nor a CR, it is output forthwith (**lines 301-309**). If the right bracket prompt is being printed (**line 304**), flow resumes at **lines 349-350**, where registers are saved and files are tested. If no file is open (**line 352**), registers are restored (**line 363**) and the immediate mode prompt is echoed (**line 364**). On finding one or more files open, **lines 355-364** flush buffered data, clear the file output flag, set true I/O handlers, print FILE(S) STILL OPEN, restore registers and echo the right bracket.

If file output is active and a CR is found, **lines 313-314** check PROMPT ($33) for consecutive CRs (see the explanation below). Whereas a single CR is written directly to the file buffer (**line 316**), two CR's in sequence flush the file buffer (**line 317**), clear the file output flag (**line 318**), and print the CR to the screen or printer rather than to the file buffer (**line 320**).

If file output is not active (**line 295**), control passes to **lines 325-327**, where the active file and/or prefix input are tested. If it's found and if the input token is being processed in **lines 332-335** (token handling is described below), the echo of the question mark prompt is suppressed by **lines 368-375**. Again, this double-CR handling mystifies me.

If neither file output nor file/prefix input is active (**line 328**), **lines 339-343** echo all but the immediate mode Applesoft prompt character.

TRACE PROCESSING

When OUTRACE output is confirmed in VSTATE4 (**line 288**), control passes to BITRACE (**lines 381-517**), which processes Applesoft tokens and controls tracing of Applesoft and ProDOS BASIC statements. Entry is always at **line 383** where the low-order byte of the text pointer is incremented. If a page boundary is encountered (**line 384**), **lines 381-382** increment the high-order byte. **Lines 388-389** set PROMPT equal to $FF. If this value is unchanged when the STATE 4 handler regains control, consecutive nontokens are flagged, and **lines 313-314** use this condition to detect successive CRs.

After restoring the stack pointer (the S-Register) to its value on entry to NEWSTT (**lines 46-47** of **Listing 2**), which effectively pops two addresses from the stack, DTRACE, the true BI trace flag, is tested (**line 398**). If it is active (**line 399**), control passes to **lines 494-504**, where the trace character and line number are printed, and program flow is routed back to **line 403**.

If DTRACE is inactive, program flow falls to **lines 403-404**, where the next program character is obtained. If a token is found (**line 405**), processing resumes at **line 436**. If the end-of-line marker (zero) is picked up (**line 406**), a branch to **lines 429-430** conditions the processor status byte (P-Register) and flow jumps to one instruction beyond GETSTT (**line 87** in **Listing 2**). If a nontoken is located, **lines 410-420** are responsible for the automatic garbage collection that is such a convenient feature of ProDOS BASIC.

STRINGS ($BE49) is an arbitrary counter that is decremented each time a nontoken is picked up. Every so often (i.e., when STRINGS equals zero), free space is tested. If three pages or more are free, garbage collection is deemed unnecessary. Less than three pages of unused space triggers flushing of the temporary (general purpose) data buffer, full-scale garbage collection and reset of STRINGS. In either circumstance, flow resumes at **lines 424-425**, where the same nontoken is obtained and control is routed to Applesoft ROM (**lines 429-430**).

Token processing (**line 436**) is the start of the real fun. Since the BI has intercepted Applesoft statement processing, it must pro-

TABLE 1: Stack Contents When Applesoft Is Tracing

Sequence	Location	Instruction	S-Register	Stack Contents	Stack Address
4	$9E2C	(BI entry)	$F4	-	$1F4
			$F5	$3C	$1F5
3	$DB3A	JSR $FDED	$F6	$DB	$1F6
			$F7	$12	$1F7
2	$D810	JSR $DB57	$F8	$D8	$1F8
1	$D7D2	(NEWSTT entry)	$F8	-	$1F8

vide reentry to NEWSTT (as noted above in **lines 429-430**). Before it does so, however, the BI imposes its will on several tokens. A table of Applesoft statement tokens is kept by the BI (APTKNTBL, **lines 569-590**). Tokens that are handled normally are stored in the usual negative ASCII format. Tokens that require special processing are replaced by branch offsets (positive numbers), the nature of which we shall see shortly.

Here's how it works. **Lines 436-437** store the token in PROMPT and convert the token to an index into APTKNTBL. **Line 438** fetches the table entry. If it is negative (i.e., in the normal format) in **line 439**, Applesoft is reentered. A positive table entry is used to modify (**line 440**) the branch instruction in **line 441**. For example, if the LIST token ($BC) is encountered, table entry $13 (**line 589**) is placed at location $9EFD (**line 441**). The branch now goes to **line 456**, where the LIST token is processed. Some programmers eschew the use of self-modifying code. I like it.

Individual tokens are handled as described below:

PRINT and IF Tokens — <CTRL>D following a PRINT statement is the signal for the BI to anticipate a command. When a PRINT token is found, a zero is stuffed into PROMPT (**line 446**), memory of the prior character is wiped out (**line 447**), STATE D is set (**lines 448-451**), and **line 452** branches to **line 482** where the token is restored to the Accumulator and Applesoft is reentered. The next output will check for <CTRL>D via the STATE D handler. I don't know why the IF statement requires similar processing.

LIST and CALL Tokens — When LIST and CALL are encountered, PROMPT is flagged with the values 1 and 2, respectively, and Applesoft is accessed (**lines 456-464**).

LET Token — **Lines 468-469** restore the token to the Accumulator and jump to the code that checks for garbage collection.

TRACE and NOTRACE Tokens — The BI trace flag is set by storing the TRACE token ($9B) in DTRACE (**line 473**) and branching (**line 474**) to **lines 480-481**, where the value $4A is stuffed into the Applesoft trace flag. On the surface, this would seem to clear TRCFLG (a no-no!), but actually, once the Applesoft trace handler rotates a set carry into bit 7 of TRCFLG, the familiar $A5 will be present. After restoring the token to the Accumulator, Applesoft is entered (**lines 482-483**).

Following the NOTRACE command, DTRACE is cleared by receiving the branch offset ($28) (**line 478**). **Line 479** converts NOTRACE to TRACE so that TRCFLG is

continued on page 122

LISTING 1: BI Output Processing
Note: Don't enter this code. It already exists in the BI.

```
                        1   ********************************************
                        2   *                                          *
                        3   *            OUTPUT PROCESSING             *
                        4   *  ProDOS BASIC INTERPRETER version 1.1    *
                        5   *      [BI 1.Ø addresses bracketed]        *
                        6   *                                          *
                        7   *            Interpreted by                *
                        8   *            Sandy Mossberg                *
                        9   *                                          *
                        1Ø  ********************************************
                        11
                        12  * Merlin-Pro Assembler
                        13
                        14  * General Equates:
                        15
                        16  CH       =     $24        ;Column position of cursor
                        17  PROMPT   =     $33        ;Prompt character
                        18  CSWL     =     $36        ;Output hook
                        19  KSWL     =     $38        ;Input hook
                        2Ø  STREND   =     $6D        ;Bottom of arrays
                        21  FRETOP   =     $6F        ;Bottom of strings
                        22  MEMSIZ   =     $73        ;HIMEM
                        23  CURLIN   =     $75        ;Current line number
                        24  TXTPTR   =     $B8        ;Apsoft text pointer
                        25  ERRNUM   =     $DE        ;Error code number (Applesoft)
                        26  TRCFLG   =     $F2        ;Apsoft trace flag
                        27  REMSTK   =     $F8        ;Stack pointer (Applesoft)
                        28  STACK    =     $1ØØ       ;Stack
                        29  INBUF    =     $2ØØ       ;Input buffer
                        3Ø  COUT     =     $FDED      ;Output char via CSW hook
                        31
                        32  * BI Proper Equates:
                        33
                        34  SETIOTRU =     $9AØØ      ;Set true I/O handlers [same]
                        35  ERROR    =     $9AEE      ;Error handler [$9B22]
                        36  REGSAV   =     $9F62      ;Save entry registers [$9F99]
                        37  REGRST   =     $9F6C      ;Restore entry registers [$9FA3]
                        38  SETSTATE =     $9F76      ;Set STATE handlers [$9FAD]
                        39  FRECMD   =     $AØ44      ;FRE command [$AØ7B]
                        4Ø  SYNTAX   =     $A677      ;Parse command line [$A6B4]
                        41  DOSOUT   =     $B7F1      ;BI output intercept [$B84B]
                        42  STATED   =     $B8Ø3      ;STATE D output handler [$B85D]
                        43  TXBUF    =     $BCBD      ;Text buffer [same]
                        44
                        45  * Applesoft ROM Equates:
                        46
                        47  OUTRACE  =     $D81Ø      ;Output trace char
                        48  GETSTT   =     $D81D      ;Execute Apsoft statement
                        49  LINPRT   =     $ED24      ;Print decimal of A,X
                        5Ø
                        51  * BI Global Page Equates [same]:
                        52
                        53  PRINTERR =     $BEØC      ;Print error message
                        54  VECTOUT  =     $BE3Ø      ;True output handler
                        55  VECTIN   =     $BE32      ;True input handler
                        56  VDOSIO   =     $BE34      ;BI intercepts
                        57  VSYSIO   =     $BE38      ;Current state handlers
                        58  PREGX    =     $BE3F      ;Save X-reg
                        59  PREGY    =     $BE4Ø      ;Save Y-reg
                        6Ø  DTRACE   =     $BE41      ;BI trace flag
                        61  IFILACTV =     $BE44      ;Input file active (if MI)
                        62  OFILACTV =     $BE45      ;Output file active (if MI)
                        63  PFXACTV  =     $BE46      ;Prefix input active (if MI)
                        64  STRINGS  =     $BE49      ;Counter for free string space
                        65  TBUFPTR  =     $BE4A      ;Temp buffer char count (WRITE)
                        66  INPTR    =     $BE4B      ;Index to command line
                        67  CHRLAST  =     $BE4C      ;Last char output
                        68  OPENCNT  =     $BE4D      ;Number of open nonEXEC files
                        69  GOSYSTEM =     $BE7Ø      ;Call MLI
                        7Ø  RWCOUNT  =     $BED9      ;R/W request.count parameter
                        71
                        72  * Special Equates:
                        73
                        74  input    =     $84        ;INPUT token
                        75  ctld     =     $84        ;CTL-D (negative ASCII)
                        76  cr       =     $8D        ;CR (negative ASCII)
                        77  space    =     $AØ        ;Space
                        78  WRITE    =     $CB        ;MLI WRITE call
                        79
                        8Ø  *-------------------------------
                        81  * STATE Ø Output Handler (Immediate Mode) [same]:
                        82  *-------------------------------
                        83          ORG   $9A2F
                        84
                        85  * Check if Applesoft is tracing:
                        86
                        87  VIDINCPT
9A2F: CD 61 9F          88  VSTATEØ  CMP   TRACECHR
9A32: DØ 2Ø             89          BNE   :1          ;Trace char not being output
9A34: 8E 3F BE          9Ø          STX   PREGX       ;Save entry X
9A37: BA                91          TSX               ;Use stack pointer as offset
9A38: BD Ø3 Ø1          92          LDA   STACK+3,X   ;Check stack for Apsoft
9A3B: C9 12             93          CMP   #OUTRACE+2  ; trace printing
9A3D: DØ 2F             94          BNE   :3          ;Not trace printing
```

```
9A3F: BD 04 01   95            LDA   STACK+4,X
9A42: C9 D8      96            CMP   #>OUTRACE+2
9A44: D0 28      97            BNE   :3          ;Not trace printing
                 98
                 99    * Applesoft is tracing:
                100
9A46: A2 04     101            LDX   #4
9A48: 20 76 9F  102            JSR   SETSTATE    ;Set deferred mode
9A4B: AD 61 9F  103            LDA   TRACECHR    ;Restore entry A
9A4E: AE 3F BE  104            LDX   PREGX       ;Restore entry X
9A51: 4C F1 B7  105            JMP   DOSOUT      ;Output the trace char
                106
                107    * Applesoft not tracing:
                108
9A54: CD 4C BE  109    :1      CMP   CHRLAST     ;Same as last char output?
9A57: 8D 4C BE  110            STA   CHRLAST     ;Remember current char
9A5A: D0 18     111            BNE   CHAROUT     ;Output different char
                112
                113    *  -> Check for consecutive CRs:
                114
9A5C: C9 8D     115            CMP   #cr         ;Two consecutive CRs?
9A5E: D0 14     116            BNE   CHAROUT     ;No, so output char
                117
                118    *  -> Consecutive CRs found:
                119
9A60: A5 24     120            LDA   CH          ;If not start of line,
9A62: D0 05     121            BNE   :2          ; output 2nd CR
9A64: AD BD BC  122            LDA   TXBUF       ;At start of line. Suppress
9A67: C9 3F     123            CMP   #'?'        ; 2nd CR if command
9A69: A9 8D     124    :2      LDA   #cr         ; not pending
9A6B: B0 07     125            BCS   CHAROUT     ;Output 2nd CR
9A6D: 60        126            RTS               ;Ignore CR
                127
                128    * Trace character being output; Applesoft not tracing:
                129
9A6E: AE 3F BE  130    :3      LDX   PREGX       ;Restore entry X
9A71: AD 61 9F  131            LDA   TRACECHR    ;Prepare to output trace char
                132    *--------------------------------
                133    * Echo Output Character [same]:
                134    *--------------------------------
                135    * Output the character:
                136
9A74: 20 00 9A  137    CHAROUT JSR   SETIOTRU    ;Set true I/O handlers
9A77: 20 ED FD  138            JSR   COUT        ;Output char
                139
                140    * Check for CR:
                141
9A7A: C9 8D     142            CMP   #cr
9A7C: D0 0F     143            BNE   SAVIOTRU    ;CR not output
                144
                145    * CR found:
                146
9A7E: A5 F2     147            LDA   TRCFLG
9A80: C9 A5     148            CMP   #$A5
9A82: F0 07     149            BEQ   :1          ;Normal value in trace flag
9A84: 8D 41 BE  150            STA   DTRACE      ;Set/clear for immediate mode
9A87: A9 A5     151            LDA   #$A5        ;Restore normal value
9A89: 85 F2     152            STA   TRCFLG      ; in trace flag
9A8B: A9 8D     153    :1      LDA   #cr         ;Restore A
                154    *--------------------------------
                155    * Save True I/O Handlers in BI global page [same]:
                156    *--------------------------------
9A8D: 48        157    SAVIOTRU PHA
9A8E: A5 39     158            LDA   KSWL+1      ;Input MSB
9A90: 8D 33 BE  159            STA   VECTIN+1
9A93: A5 38     160            LDA   KSWL        ;Input LSB
9A95: 8D 32 BE  161            STA   VECTIN
9A98: A5 37     162            LDA   CSWL+1      ;Output MSB
9A9A: 8D 31 BE  163            STA   VECTOUT+1
9A9D: A5 36     164            LDA   CSWL        ;Output LSB
9A9F: 8D 30 BE  165            STA   VECTOUT
9AA2: 68        166            PLA
                167    *--------------------------------
                168    * Put Intercepts Into CSW/KSW [same]:
                169    *--------------------------------
9AA3: 48        170    SETINCPT PHA
9AA4: AD 34 BE  171            LDA   VDOSIO      ;Output LSB
9AA7: 85 36     172            STA   CSWL
9AA9: AD 35 BE  173            LDA   VDOSIO+1    ;Output MSB
9AAC: 85 37     174            STA   CSWL+1
9AAE: AD 36 BE  175            LDA   VDOSIO+2    ;Input LSB
9AB1: 85 38     176            STA   KSWL
9AB3: AD 37 BE  177            LDA   VDOSIO+3    ;Input MSB
9AB6: 85 39     178            STA   KSWL+1
9AB8: 68        179            PLA
9AB9: 60        180            RTS
                181    *--------------------------------
                182    * STATE D Output Handler (Check CTL-D) [$9DD2]:
                183    *--------------------------------
                184            ORG   $9DA3
                185
                186    * Check CTL-D after PRINT token:
                187
9DA3: 20 62 9F  188    VSTATED JSR   REGSAV      ;Save A,X,Y
9DA6: C9 84     189            CMP   #ctld
9DA8: D0 17     190            BNE   :1          ;CTL-D not found
                191
                192    * CTL-D found so prepare to assemble command line:
```

continued on next page

LISTING 1: BI Output Processing *(continued)*

```
                        193
9DAA: 20 F4 9F          194           JSR   WRBUFDA1    ;Write buffered data
9DAD: 8D 4B BE          195           STA   INPTR       ;Set at start of command line
9DB0: 8D 44 BE          196           STA   IFILACTV    ;Set file input inactive
9DB3: 8D 45 BE          197           STA   OFILACTV    ;Set file output inactive
9DB6: 8D 46 BE          198           STA   PFXACTV     ;Set prefix input inactive
9DB9: A2 08             199           LDX   #8
9DBB: 20 76 9F          200           JSR   SETSTATE    ;Set STATE 8
9DBE: 4C 6C 9F          201           JMP   REGRST      ;Restore A,X,Y
                        202
                        203  * CTL-D not found so return to deferred mode:
                        204
9DC1: A2 04             205  :1       LDX   #4
9DC3: 20 76 9F          206           JSR   SETSTATE    ;Set STATE 4
9DC6: 20 6C 9F          207           JSR   REGRST      ;Restore A,X,Y
9DC9: 4C F1 B7          208           JMP   DOSOUT      ;Output the char
                        209  *------------------------------
                        210  * STATE 8 Output Handler (Assemble Command Line) [$9DFB]:
                        211  *------------------------------
                        212  * Save char in command line buffer (INBUF):
                        213
9DCC: 20 62 9F          214  VSTATE8  JSR   REGSAV      ;Save A,X,Y
9DCF: AE 4B BE          215           LDX   INPTR       ;Set index to command line
9DD2: 9D 00 02          216           STA   INBUF,X     ;Save char in command line
9DD5: C9 8D             217           CMP   #cr
9DD7: F0 0E             218           BEQ   V8CREX      ;CR found
                        219
                        220  * Line not completed:
                        221
9DD9: EE 4B BE          222           INC   INPTR       ;Bump index
9DDC: D0 05             223           BNE   V8NOCREX    ;Go to exit
9DDE: A9 10             224           LDA   #$10        ;Set for SYNTAX ERROR
9DE0: 4C EE 9A          225  GOERROR2 JMP   ERROR       ;Go to error handler
9DE3: AE 3F BE          226  V8NOCREX LDX   PREGX
9DE6: 60                227           RTS
                        228
                        229  * Line completed (CR found):
                        230
9DE7: E8                231  V8CREX   INX
9DE8: CA                232           DEX
9DE9: F0 0B             233           BEQ   :1          ;Empty line found
9DEB: 20 00 9A          234           JSR   SETIOTRU    ;Set true I/O handlers
9DEE: 20 77 A6          235           JSR   SYNTAX      ;Process command line
9DF1: B0 ED             236           BCS   GOERROR2    ;Error
9DF3: 20 8D 9A          237           JSR   SAVIOTRU    ;Restore intercepts
9DF6: A2 04             238  :1       LDX   #4          ;Processing completed so
9DF8: 20 76 9F          239           JSR   SETSTATE    ; set STATE 4
9DFB: 4C 6C 9F          240           JMP   REGRST      ;Restore A,X,Y
                        241  *------------------------------
                        242  * Write Character to Temporary Buffer [$9E2D]:
                        243  *------------------------------
                        244  * Check for desired Applesoft token:
                        245
9DFE: 8C 40 BE          246  WRTBFCHR STY   PREGY       ;Save Y
9E01: A4 33             247           LDY   PROMPT
9E03: F0 0C             248           BEQ   :1          ;PRINT or IF (prompt=0)
9E05: 88                249           DEY
9E06: F0 09             250           BEQ   :1          ;LIST (prompt=1)
9E08: 88                251           DEY
9E09: F0 06             252           BEQ   :1          ;CALL (prompt=2)
9E0B: AC 40 BE          253           LDY   PREGY       ;Restore Y
9E0E: 4C 74 9A          254           JMP   CHAROUT     ;Echo char
                        255
                        256  * PRINT, IF, LIST or CALL Applesoft token found:
                        257
                        258  *  -> Store character in temporary buffer:
                        259
9E11: AC 4A BE          260  :1       LDY   TBUFPTR     ;Get index to buffer
9E14: 29 7F             261           AND   #$7F        ;Convert char to pos ASCII
9E16: 91 73             262           STA   (MEMSIZ),Y  ;Store char in buffer
9E18: AC 40 BE          263           LDY   PREGY       ;Restore Y
9E1B: EE 4A BE          264           INC   TBUFPTR     ;Bump buffer index and exit if
9E1E: D0 0B             265           BNE   RTS3        ; buffer not full (256 bytes)
                        266
                        267  *  -> Write full buffer to disk:
                        268
9E20: 20 62 9F          269           JSR   REGSAV      ;Save A,X,Y
9E23: 20 EE 9F          270           JSR   WRBUFDAT    ;Write buffer to disk
9E26: B0 B8             271           BCS   GOERROR2    ;Error
9E28: 20 6C 9F          272           JSR   REGRST      ;Restore A,X,Y
9E2B: 60                273  RTS3     RTS
                        274  *------------------------------
                        275  * STATE 4 Output Handler [$9E5B]:
                        276  *------------------------------
                        277  * Check if Applesoft is tracing:
                        278
9E2C: CD 61 9F          279  VSTATE4  CMP   TRACECHR    ;Site of THEN FLASH patch
9E2F: D0 18             280           BNE   :2          ;Trace char not being output
9E31: 8E 3F BE          281           STX   PREGX       ;Save entry X
9E34: BA                282           TSX
9E35: BD 03 01          283           LDA   STACK+3,X   ;Check stack for Apsoft
9E38: C9 12             284           CMP   #OUTRACE+2  ; trace printing
9E3A: D0 07             285           BNE   :1          ;Not trace printing
9E3C: BD 04 01          286           LDA   STACK+4,X
9E3F: C9 D8             287           CMP   #>OUTRACE+2
9E41: F0 73             288           BEQ   BITRACE1    ;Trace printing active
```

```
                    289
                    290  * Applesoft not tracing. Check file output activity:
                    291
9E43: AD 61 9F      292  :1      LDA   TRACECHR   ;Restore entry A
9E46: AE 3F BE      293          LDX   PREGX      ;Restore entry X
9E49: 2C 45 BE      294  :2      BIT   OFILACTV
9E4C: 10 1E         295          BPL   :4         ;File output not active
                    296
                    297  * File output active:
                    298
                    299  *  -> Check Applesoft prompt character:
                    300
9E4E: C9 DD         301          CMP   #"]"
9E50: D0 04         302          BNE   :3         ;] not being output
9E52: C5 33         303          CMP   PROMPT     ;If Apsoft prompt being
9E54: F0 30         304          BEQ   :6         ; output, check open files
                    305
                    306  *  -> Check CR:
                    307
9E56: C9 8D         308  :3      CMP   #cr        ;If CR not being output, write
9E58: D0 A4         309          BNE   WRTBFCHR   ; char to temporary buffer
                    310
                    311  *  -> CR being output:
                    312
9E5A: A5 33         313          LDA   PROMPT     ;If prompt is -1 ($FF), 2nd
9E5C: C9 FF         314          CMP   #-1        ; consecutive CR being output
9E5E: A9 8D         315          LDA   #cr        ;Restore A
9E60: 90 9C         316          BCC   WRTBFCHR   ;Write 1st CR
9E62: 20 F4 9F      317          JSR   WRBUFDA1   ;In case of error flush buffer
9E65: 8D 45 BE      318          STA   OFILACTV   ; & flag file output inactive
9E68: A9 8D         319          LDA   #cr        ;Restore A
9E6A: D0 33         320          BNE   :8         ;Always output 2nd CR to device
                    321
                    322  * File output not active. Check active
                    323  * file input and/or prefix input:
                    324
9E6C: 48            325  :4      PHA
9E6D: AD 44 BE      326          LDA   IFILACTV   ;Test active file input
9E70: 0D 46 BE      327          ORA   PFXACTV    ;Test active prefix input
9E73: 10 08         328          BPL   :5         ;File/prefix input inactive
                    329
                    330  * File input and/or prefix input active:
                    331
9E75: A5 33         332          LDA   PROMPT
9E77: 09 04         333          ORA   #4
9E79: C9 84         334          CMP   #input     ;If INPUT token found,
9E7B: F0 25         335          BEQ   :9         ; suppress echo of prompt
                    336
                    337  * Check Applesoft prompt character:
                    338
9E7D: 68            339  :5      PLA
9E7E: C9 DD         340          CMP   #"]"       ;If char is not ],
9E80: D0 1D         341          BNE   :8         ; echo char
9E82: C5 33         342          CMP   PROMPT     ;If char is ] but ] is not
9E84: D0 19         343          BNE   :8         ; prompt, echo char
                    344
                    345  * Applesoft immediate mode prompt being output:
                    346
                    347  *  -> Check open file(s):
                    348
9E86: 20 62 9F      349  :6      JSR   REGSAV     ;Save A,X,Y
9E89: AD 4D BE      350          LDA   OPENCNT
9E8C: F0 0E         351          BEQ   :7         ;None open so echo char
                    352
                    353  *  -> File(s) open:
                    354
9E8E: 20 F4 9F      355          JSR   WRBUFDA1   ;Write data to buffer
9E91: 8D 45 BE      356          STA   OFILACTV   ;Set output file inactive
9E94: 20 00 9A      357          JSR   SETIOTRU   ;Set true I/O handlers
9E97: A9 15         358          LDA   #$15
9E99: 20 0C BE      359          JSR   PRINTERR   ;Print FILE(S) STILL OPEN
                    360
                    361  * Echo output character:
                    362
9E9C: 20 6C 9F      363  :7      JSR   REGRST     ;Restore A,X,Y
9E9F: 4C 74 9A      364  :8      JMP   CHAROUT    ;Echo output char
                    365
                    366  * No echo exit:
                    367
9EA2: 68            368  :9      PLA
9EA3: C9 8D         369          CMP   #cr
9EA5: D0 03         370          BNE   :10        ;CR not found
9EA7: CD 4C BE      371          CMP   CHRLAST    ;Prior char CR?
9EAA: 8D 4C BE      372  :10     STA   CHRLAST    ;Remember present char
9EAD: D0 02         373          BNE   RTS4
9EAF: 85 33         374          STA   PROMPT     ;Flag consecutive CRs
9EB1: 60            375  RTS4    RTS
                    376  *-------------------------------
                    377  * BI Trace Processing [$9EE1]:
                    378  *-------------------------------
                    379  * Bump Applesoft text pointer:
                    380
9EB2: E6 B9         381  BITRACE  INC   TXTPTR+1
9EB4: D0 04         382           BNE   BITRACE2
9EB6: E6 B8         383  BITRACE1 INC   TXTPTR
9EB8: F0 F8         384           BEQ   BITRACE
                    385
```

continued on page 124

not cleared.

RESUME Token — Lines 487-490 clear the Applesoft ONERR code, restore the token to the Accumulator and branch to an Applesoft entry vector.

INVERSE and NORMAL Tokens — Lines 513-514 stuff the pound sign into the trace character image, and **lines 515-516** restore the token to the Accumulator and reenter Applesoft BASIC.

FLASH Token — When the FLASH token is found, the value $E3 is stored in the image of the trace character (**lines 508-509**). The rationale can be found in OUTDO at **line 118** of **Listing 2**. When FLASH mode is active, ORMASK holds the value $40 (**lines 180-182** of **Listing 2**), and the pound sign is ORed with $40 and becomes $E3. Whenever OUTDO is called to print the trace character with FLASH in effect, the trace character is represented by $E3 rather than $A3. For the tests at **lines 88 and 279** to remain valid under this circumstance, the trace character image must now contain $E3. Please remember this theory for future reference.

STATE D

The sole purpose of STATE D is to search for < CTRL > D following a PRINT token. If the DOS alert character is not found (**line 190**), STATE 4 is reset (**lines 205-206**) and VSTATE4 is accessed (**line 208**). If < CTRL > D is located, a BI command is pending, in which case the temporary buffer is cleared, the command line index is zeroed, file/prefix I/O flags are cleared, and STATE 8 is activated (**lines 194-200**).

STATE 8

When a BI command has been identified, VSTATE8 assembles the command in the input buffer where it can be validated and processed. INPTR (**line 215**) holds the buffer position, and **line 216** stores the command character in the input buffer. If the end of the command (i.e., a CR) is not found, INPTR is incremented (**line 222**) and the routine is exited (**lines 226-227**). If the command line contains more than 256 characters, **lines 224-225** produce a SYNTAX ERROR message. On completion of the command (**line 218**), flow passes to **lines 231-232**, where a noncommand (i.e., < CTRL > D followed by no command) is ignored (**line 233**) or a pending command is processed (**line 235**). Following command execution, a set carry indicates an error condition (**line 236**). On Carry Clear, STATE 4 is reinstated (**lines 238-239**).

FILE OUTPUT

When a file is being written, some output is directed to the file itself (e.g., after PRINT statements) and some output goes to the screen or printer. If a PRINT, CALL or LIST token is found (these were flagged by BITRACE when special tokens were processed) (**lines 247-252**), output is held in the temporary (general purpose) buffer; otherwise, the character is echoed (**line 254**). Buffered characters are stored in positive ASCII format (**lines 260-262**). When the buffer contains 256 bytes, it is written to disk (**lines 269-273**).

WRBUFDAT does the actual writing. If entered at **line 529**, 256 bytes are copied from the temporary buffer to disk. Entry at **line 535** writes the number of bytes in the A and Y Registers to disk and handles errors. Entry at **line 545** flushes the buffer (TBUFPTR holds the number of stored bytes) to disk. **Lines 548-551** set the WRITE parameter list for the correct number of bytes to be written and call GOSYSTEM, the BI global page subroutine that handles calls to the machine language interface. After resetting TBUFPTR to zero (**lines 553-554**), the subroutine exits with a zero in the Accumulator. As usual, a set carry flags an error.

TRACING AN ELUSIVE BUG

Since the inception of ProDOS, I and others have noted a strange bug in the trace mechanism. An Applesoft program might be functioning well when, all of a sudden, without any apparent rhyme or reason, tracing would be activated for a brief period of time and then would stop as inexplicably as it started. This occurred without a single TRACE command found in the resident program. The addition of NOTRACE would not abolish the aberration. Strange!

As I embarked upon writing this installment of D/L, I resolved that I would root out the bug. First, however, the problem had to be defined. Fortuitously, in the April 1985 issue of *Nibble* (Vol. 6/No. 4), a letter to the editor from J.R. Wakefield presented a short program called FLASHER that illustrated this very bug. It looked like this:

```
10 GET A$:PRINT
20 IF A$ = "A" THEN FLASH:PRINT "
   ERROR":GOTO 40
30 FLASH:PRINT "OK"
40 NORMAL:GOTO 10
```

The four-liner contains no TRACE command, yet tracing occurs whenever the upper-case A is entered. It is apparent that should the IF condition (**line 20**) be satisfied, the combination of THEN and FLASH turns on tracing. When the NORMAL statement is executed, tracing stops. Experimenting confirms the culpability of this nefarious coalition of keywords. The suggestion by the editor to use POKE 50,127 instead of FLASH was valid, but I was not about to walk away with my tail between my legs. No siree!

In debugging, I use several techniques. If looking at code does not solve my problem, I use the "bell" technique to test various segments (this is explained below). When all else fails, I use BUGBYTER, the debugger on the ProDOS Assembler Tools disk. I used this superb debugger long before Apple Computer purchased it from Computer Advanced Ideas.

After racking my brain for a good hour, I could find no apparent flaw in the STATE handlers or BI trace handler. I theorized that the problem must reside in the interaction between Applesoft and ProDOS BASIC.

My next step was to see how the FLASH token was processed by the BI. The bell method is a simple debugging trick: you place a call to the bell routine in a segment of code. The bell will sound if program flow passes through the target area. I entered the System Monitor (CALL -151) and made the following patches (BI 1.0 addresses are in brackets):

```
9F54:4C 4C BB JMP $BB4C [9F83:4C BA BB]
BB4C:20 3A FF JSR BELL  [BBBA:20 3A FF]
     A9 E3    LDA #$E3  [      A9 E3   ]
     4C 5A 9F JMP $9F5A [      4C 89 9F]
```

$BB4C-$BC79 is an unused space within BI version 1.1 ($BBBA-$BC77 is free in version 1.0). This code merely provides a bell sound whenever the BI processes the FLASH token (**lines 508-509** and **514-515** of **Listing 1**).

I reentered BASIC and again ran FLASHER. To my delight, the bell rang (i.e., FLASH was processed by the BI) when any key but A was pressed, but no sound was heard when the IF-THEN statement was evaluated as true (i.e., A was pressed). This proved that the FLASH in **line 30** was handled normally, but the same token in **line 20** was ignored by the BI. Very interesting.

Apparently, a true IF-THEN construction caused a token following THEN to bypass the BI. I tested this hypothesis using the following program:

```
100 D$ = CHR$(4)
110 ONERR GOTO 500
120 PRINT D$"DELETE TESTFILE"
130 PRINT PEEK(222):END
500 PRINT D$"CREATE TESTFILE"
510 IF NOT B THEN RESUME
```

For the program to function properly, TESTFILE cannot be a file on your disk. When **line 120** is executed, TESTFILE is not found and ONERR flips control to **line 500**, where TESTFILE is created. The RESUME statement passes flow back to the point at which the error occurred (i.e. **line 120**). When Applesoft processes RESUME, the error code location ($DE = 222) is not cleared. When the BI preprocesses RESUME, however, this location is zeroed (**lines 487-488** of **Listing 1**). Thus, if the BI gets hold of RESUME, **line 130** prints a zero before ending. If the BI fails to process RESUME, error number 6 (FILE NOT FOUND) is output. It all hinges on whether a true IF-THEN complex will skip RESUME, as it apparently did with FLASH in

"We will match any mail order price in this issue."

William F. Gollan
President, BCP

We'll support, service and warranty everything we sell and stand behind it 100%. We won't charge you for shipping or handling, nor will we penalize you for using a credit card.

MODEMS

D.C. Hayes		***Prometheus***	
Smart 300 IIc w/sc & cable	179.95	Pro 1200	**309.95**
Micro Modem IIe w/sc	149.95	Pro 1200 A w/sc	**299.95**
Smart 1200	**384.95**	Pro 300C	**CALL**
Smart Modem 2400	**594.95**	***Zoom***	
Transit w/128k	**289.95**	Zoom Micro IIe + w/sc	159.95

PERIPHERALS & ACCESSORIES

Apple		MD 2 DS/DD (10)	**26.95**
II E's & II C's & Macs	**CALL**	***Kensington***	
AST		Dust Image 10"	**9.95**
Multi I/O	**164.95**	Printer Stand	**19.95**
Checkmate Tech		System Saver	64.95
Multiram II E 64K +80	**129.95**	***Koala Technologies***	
Multiram II E 256K +80	**169.95**	Koala Pad	74.95
Multiram II E 512K +80	**219.95**	***Joysticks***	
Multiram II C w/256K	**239.95**	Kraft	39.95
Multiram II C w/512K	**289.95**	Kraft Quickstick	**54.95**
Curtis		Hayes Mach III's	39.95
Diamond	**34.95**	***Ribbons***	
Emerald	39.95	Black & Multi Colors	**CALL**
Ruby	59.95	***Titan***	
Static Mat	**27.95**	Accellorator	**224.95**
Disks Verbatim		64k Ram	144.95
Bonus SS/DD (10)	12.95	128k Ram	164.95
Bonus DS/DD (10)	16.95	Neptune 192k + 80	239.95
Disks Maxell			
MD 1 SS/DD (10)	**19.95**		

GRAPHIC INTERFACES & BUFFERS

Orange Micro		***Textprint***	
Grappler +	84.95	Print It	**129.95**
Grappler + w/64k	174.95	***Thirdware***	
Grappler + RS-232C	84.95	Fingerprint +	**94.95**
Grappler C	**84.95**	Fingerprint + IIC	**94.95**
Hot Link IIc	**49.95**		

PRINTERS DOT MATRIX

Apple		LQ 1500	**929.95**
Imagewriter II 10"	**474.95**	***Okidata***	
Imagewriter 15"	**509.95**	ML 192	**359.95**
Citizen		ML 192 Image.	**389.95**
MSP 10 160	**274.95**	ML 193	**489.95**
MSP 15 160	**374.95**	ML 84	**654.95**
MSP 20 200	**344.95**	***Star Micronics***	
MSP 25 200	**494.95**	SG 10	239.95
Epson America		SD 10	**359.95**
LX 80	239.95	SR 10	**499.95**
LX 80 F/T	**274.95**	***Toshiba***	
FX 85	**369.95**	P 1340	**529.95**
FX 185	**499.95**		

BUSINESS SOFTWARE

ALS		***Kensington***	
List Handler	**54.95**	Format II	94.95
Word Handler	**54.95**	***Living Videotext***	
Spell Handler	**39.95**	Think Tank	**84.95**
Apple		***Megahaus***	
Appleworks	199.95	MegaWorks	79.95
Arrays		Report Works	**CALL**
Home Accountant	**47.95**	***Microsoft***	
BPI Software		Multiplan	**59.95**
A/R	**229.95**	***Monogram***	
G/L	**229.95**	Dollars & Cents	64.95
P/R	**229.95**	Dollars & Cents IIc	74.95
A/P	**229.95**	Forcast	44.95
I/C	**229.95**	***Penguin***	
Broderbund		Home Data Mng.	**29.95**
Bank Street Writer	44.95	Home Connection	**34.95**
Bank Street Speller	44.95	***Quark***	
Bank Street Filer	**44.95**	Catalyst	**89.95**
CBS		Word Juggler	**119.95**
Stockpac II	**94.95**	***Roger Wagner***	
Compuserve		Mouse Write	**74.95**
Subscription Kit	**27.95**	***Sensible Software***	
Dow Jones		Sensible Speller IV	74.95
Market Analyzer +	209.95	Graphics Dept.	**79.95**
Investors Workshop + IIc	**89.95**	***Sierra On Line***	
Market Microscope	**209.95**	Homeword	44.95
Market Mng.	**149.95**	Homeword Speller	34.95
Human Edge		Home Word Filer	**44.95**
Mind Prober	**344.95**		

BUSINESS SOFTWARE

Software Publishing		***Sorcim/IUS***	
PFS:1st Success	**139.95**	Super IIIA	**119.95**
PFS:Write	74.95	***SSI***	
PFS:File	74.95	Word Perfect	**99.95**
PFS:Graph	74.95	***Stoneware***	
PFS:Report	74.95	D.B. Master V 4.0 +	**179.95**
PFS:Plan	**74.95**	***United Software***	
		ASCii Express	84.95

UTILITIES & LANGUAGES SOFTWARE

Baudville		Turbo Tutorial	25.95
Actors & Actions T1	20.95	***Broderbund***	
Business T1	34.95	Dazzle Draw	37.95
Blazing Saddles	**34.95**	Print Shop	34.95
Heroes & Villans T1	20.95	Print Shop GL 1,2 (ea)	**18.95**
Pixit	34.95	Print Shop Refill	13.95
Shape Lib. 1,2,3 (ea)	**13.95**	***Central Point***	
Take 1	37.95	Copy II +	**24.95**
Beagle Brothers		***Funk***	
Alpha Plot	**27.95**	Sideways	**38.95**
Apple Mechanic	20.95	Sideways Pro DOS	**CALL**
Bag	20.95	***Microsoft***	
Basic	25.95	Basic Compiler	**234.95**
Graphics	37.95	COBOL-80	**399.95**
D-Code	27.95	Fortran-80	**119.95**
Diskquick	20.95	***Mindscape***	
DOS Boss	17.95	Mister Pixel's	
Double-Take	25.95	Cartoon Kit	**20.95**
Fat Cat	25.95	Paint Set	**20.95**
Flex Text	20.95	Color Me (Main)	**20.95**
Frame-Up	20.95	Rainbow Brite	**7.95**
GPLE	32.95	Shirt Tales	**7.95**
I.O. Silver	20.95	Hugga Bunch	**7.95**
Pro-Byter	**25.95**	Tink Tonk	**7.95**
Pronto DOS	20.95	***Penguin Software***	
Silicon Salad	17.95	Cat Graphics	24.95
Tip Disk # 1	**13.95**	Graphics Magician	37.95
Triple Dump	27.95	Magic Paintbrush	24.95
Typefaces	**13.95**	***Pinpoint***	
Utility City	17.95	Pinpoint	**45.95**
Borland International		***Terrapin***	
Turbo Pascal 3.0	**44.95**	LOGO	64.95
Turbo Tool	**37.95**		

EDUCATIONAL & PERSONAL SOFTWARE

Barrons		SAT Verbal	**27.95**
Computer SAT	**54.95**	***Learning Company***	
GMAT	**54.95**	Addition Magician	**24.95**
CBS		Gertrudes Puzzle	**29.95**
Goren's Bridge	49.95	Gertrudes Secrets	**29.95**
Mastering SAT	**64.95**	Juggle's Rainbow	**20.95**
Websters:Word Game	**19.95**	Magic Spells	**24.95**
Success Series	**24.95**	Moptown Hotel	**27.95**
Davidson Associates		Moptown Parade	**27.95**
Classmate	**32.95**	Reader Rabbit	**27.95**
Math Blaster	32.95	Robot Odyssey	34.95
Speed Reader II	44.95	Rocky's Boot	34.95
Data Disks A,B,C,D (ea)	**13.95**	Word Spinner	**24.95**
Spell It	32.95	***Meca***	
Word Attack	32.95	Managing Your $	**109.95**
Data Grade(s) (ea)	**13.95**	***Mindscape***	
Data Roots & Pref.	**13.95**	Perfect Score	**49.95**
Data SAT	**13.95**	Keyboard Cadet	**27.95**
Hayden		Lusher Profile	**27.95**
SAT Math	**27.95**	Bank St. Music	**34.95**

EDUCATIONAL & PERSONAL SOFTWARE

Mindscape		***Softsync***	
Bank St. Story B.	**27.95**	Kid Pro Quo	**27.95**
Quality		Model Diet	**25.95**
Bag of Tricks	28.95	Personal Acct.	**34.95**
Beneath Apple DOS	15.95	***Spinnaker***	
Beneath Apple Pro DOS	15.95	Story Machine	**27.95**
Random House		Alphabet Zoo	**20.95**
Charlie Brown's ABC's	**27.95**	Delta Drawing	**27.95**
Snoopy Rescue	**27.95**	Kids on Keys	**19.95**
Scarborough		Kindercomp	**19.95**
Boston Diet	**49.95**	Snooper Trooper 1,2 (ea)	**24.95**
Build a Book	27.95	In Search of	**24.95**
Build Refills 1,2 (ea)	**13.95**	***Springboard***	
Mastertype		The Newsroom	**34.95**
New Improved	27.95	***Virtual Combinatics***	
Figure's & Formulas	**19.95**	Micro Cookbook	27.95
Filer	**27.95**	Added Menue's (ea)	**9.95**
Writing Wizard	**49.95**	***Xerox/Weekly Reader***	
Run for the $	27.95	Sticky Bear	
Songwriter	27.95	ABC	27.95
Personal Net W.	49.95	Basket Bounce	**27.95**
Screenplay		Math	27.95
Ken Houston Black Jk.	**49.95**	Numbers	**27.95**
Trivia Arcade	27.95	Opposites	**27.95**
Simon & Schuster		Reading	**27.95**
Typing Tutor III	34.95	Shapes	**27.95**

GAME SOFTWARE

Activision		***Mindscape***	
Master Lands	**27.95**	Halley Project	**27.95**
Countdown to Shut.	**27.95**	Stephen King Mist	**27.95**
Road Race	**27.95**	James Bond View K.	**27.95**
Alcazar	**27.95**	Forbidden Castle	**27.95**
Ghostbuster's	**27.95**	Voodoo Island	**27.95**
H.E.R.O.	**27.95**	Racter	**29.95**
Pitfall	**27.95**	Tink's Adventure	**20.95**
Space Shuttle	**27.95**	Tonk in Buddy Bots	**20.95**
Mindshadow	**27.95**	Tuk Goes Town	**20.95**
Rock N Bolt	**27.95**	Tinka's Mazes	**20.95**
Blue Chip		Tinka's Sub. Fair	**20.95**
Barron	**29.95**	***Origin***	
Millionaire	**29.95**	Caverns Callisto	**24.95**
Squire	**34.95**	Ultima III Exodus	**37.95**
Tycoon	**29.95**	Autodvel	**CALL**
Broderbund		Ogre	**CALL**
Championship Lode R.	24.95	***Penguin***	
Choplifter	24.95	Pensate	**19.95**
Captain Goodnight	**24.95**	Quest	24.95
CBS		Swords Kadash	24.95
Murder by Doz.	24.95	Transylvania	24.95
Datasoft		Crimson Crown (New)	**24.95**
Bruce Lee	27.95	***Pryority***	
Conan	27.95	Forbidden Quest	**27.95**
Dig Dug	**20.95**	***Sierra On Line***	
PacMan	**20.95**	Frogger	24.95
Zaxxon IIc	**20.95**	King's Quest	34.95
Epyx		***Sir Tech***	
Rogue	**20.95**	Wizardry 1	34.95
Summer Games	**27.95**	Wizardry 2	24.95
Pitstop	**27.95**	Wizardry 3	27.95
Baseball	**24.95**	Wizardry 4	29.95
Summer Games II	**27.95**	***Spectrum***	
Winter Games	**CALL**	Gato	27.95
Football	**CALL**	***Spinnaker/Telarium***	
Temple Apshair	**20.95**	Amazon	**27.95**
Hayden		Dragon World	**27.95**
Holy Grail	**34.95**	Fahrenheit 451	**27.95**
Sargon III	34.95	Rendezvous w/Rama	**27.95**
Infocom		Shawdow Keep	**27.95**
Cutthroats	**27.95**	Homework Writing	**27.95**
Deadline	34.95	Homework Word	**27.95**
Enchanter	27.95	Homework Math	**21.95**
Hitchhiker's G.	27.95	***Strategic Simulations***	
Sorcerer	**29.95**	Baltic 85	24.95
Suspect	29.95	Battle Normandy	**27.95**
Witness	27.95	Carrier Force	**37.95**
Zork 1	27.95	Computer Ambush	**37.95**
Zork 2,3 (ea)	**29.95**	Computer Quarterback	**27.95**
Micro Lge. Sports		Computer Baseball	**27.95**
Baseball	27.95	Germany 85	**37.95**
Microprose		Kampfgruppe	**37.95**
F-15 Strike	24.95	Objective Kursk	**27.95**
NATO Commander	24.95	Questron	**34.95**
Solo Flight	24.95	Reforger 88	**37.95**
Crusade in Europe	**27.95**	War with Russia	**49.95**
Kennedy Approach	**CALL**	***Sublogic***	
Gunship	**CALL**	Flight Simulator II	34.95
Acro Jet	**CALL**		

MONTHLY SPECIAL

VERBATIM Data Lifes

	Retail	BCP
* Single Sided/Double Density (10)	$39.95	$21.95
* Head Cleaning Kit	19.95	FREE
* Library Case	5.95	FREE

SAVE $43.90

+2.50 Shipping

FREE SHIPPING
CONTINENTAL USA/UPS ONLY
APO/FPO WELCOME

MasterCard VISA

BUSINESS COMPUTERS
OF PETERBOROUGH
Gollan Co., Inc

766 RT. 101 W. • P.O. Box 389 • Peterborough, NH 03458

ORDERS ONLY 800-845-3003

QUESTIONS 603-924-9406

the Wakefield program. By the way, the Boolean "NOT B" (**line 510**) evaluates as true because all undefined variables are assigned a zero value, which goes to show that things do sometimes go your way. Indeed, on running the program, RESUME is never handled by the BI, as evidenced by the "6" on the screen. Removing the IF-THEN statement in **line 510** (i.e., simply having RESUME) produces the expected zero in location 222.

The success of the above test sent me poring through my disassembly of Applesoft ROM for the mechanism of the IF-THEN function. I searched the Applesoft command table for THEN but learned that no handler exists for this token. The IF statement handler is disassembled in **lines 131-150** of **Listing 2**. After evaluating the formula after IF (**line 131**), the character following the formula is obtained (**line 132**). If GOTO is found, flow drops to **line 137**. If THEN is found by the subroutine SYNCHR (**lines 135-136**), flow resumes at **line 137**. If neither GOTO nor THEN is encountered, SYNCHR produces the appropriate error message.

At **lines 137-138**, if the floating-point accumulator (FAC) is zero, the IF-THEN statement was false and the remainder of the line is bypassed (**lines 142-143**). When IF-THEN is true, flow passes to **line 147**, which fetches the character following THEN or GOTO. CHRGOT exits with a clear carry if a decimal digit is found or a set carry if a nondigit is picked. On finding a numeral, GOTO is inferred and control passes to the GOTO handler (**line 149**). Upon locating a nondigit, flow is routed to TYPSTT (**line 92**), the portion of NEWSTT that accesses the statement handler.

As a result of this process, any token immediately following THEN bypasses EXECUTE/GETSTT (**lines 75-78**), is not trapped by OUTRACE, and does not reach the BI. Quite fascinating, don't you think?

We have not yet explained why THEN-FLASH turns on trace mode, but we're sneaking up on it. If FLASH (following THEN) fails to be processed by the BI, the value $E3 is not placed into TRACECHR, but Applesoft still conditions ORMASK and INVFLG (**lines 174-182**). When deferred mode output next reaches OUTRACE, OUTDO uses ORMASK to convert the pound sign (value $A3) to value $E3. Since TRACECHR has not been changed from $A3 to $E3 by the BI trace processor, when VSTATE4 is reached, the branch in **line 280** of **Listing 1** is taken. The trace character (value $E3 ANDed by COUT1 with value $7F equal to a flashing pound sign) is eventually output (branches are taken at **lines 295, 328 and 341**) by CHAROUT (**line 364**). Applesoft regains control at **line 82** of **Listing 2**, where the line number is printed. Not until FLASH is disabled by NORMAL or INVERSE or until FLASH is

LISTING 1: BI Output Processing *(continued)*

```
                    386  * Set prompt for possible error detection in VSTATE4:
                    387
9EBA: A2 FF         388  BITRACE2 LDX   #-1          ;If PROMPT remains -1 then
9EBC: 86 33         389          STX   PROMPT       ; consecutive nontokens found
                    390
                    391  * Restore Applesoft NEWSTT stack pointer:
                    392
9EBE: A6 F8         393          LDX   REMSTK
9EC0: 9A            394          TXS
                    395
                    396  * Check if BI trace active:
                    397
9EC1: 2C 41 BE      398          BIT   DTRACE
9EC4: 30 73         399          BMI   DOTRACE      ;BI trace active
                    400
                    401  * Get Applesoft program character:
                    402
9EC6: A0 00         403  NXTASCHR LDY   #0
9EC8: B1 B8         404          LDA   (TXTPTR),Y ;Get char
9ECA: 30 25         405          BMI   CHKTKN       ;Token found
9ECC: F0 20         406          BEQ   GOGETST2     ;End of line found
                    407
                    408  * Check possible garbage collection:
                    409
9ECE: CE 49 BE      410  TRCFRE   DEC   STRINGS     ;Decrement counter
9ED1: D0 19         411          BNE   GOGETST1     ;If counter zero skip garbage
9ED3: A5 70         412          LDA   FRETOP+1     ;Calculate
9ED5: E5 6E         413          SBC   STREND+1     ; free space
9ED7: C9 03         414          CMP   #3           ;If 3 pages or more,
9ED9: B0 0A         415          BCS   :1           ; no problem,
9EDB: 20 F4 9F      416          JSR   WRBUFDA1     ; else write data to buffer
9EDE: 20 44 A0      417          JSR   FRECMD       ; and collect garbage
9EE1: A5 70         418          LDA   FRETOP+1     ; and recalculate
9EE3: E5 6E         419          SBC   STREND+1     ; free space
9EE5: 8D 49 BE      420  :1       STA   STRINGS      ;Save new count
                    421
                    422  * Get Applesoft program character again:
                    423
9EE8: A0 00         424          LDY   #0
9EEA: B1 B8         425          LDA   (TXTPTR),Y ;Get char
                    426
                    427  * Execute Applesoft statement:
                    428
9EEC: C9 3A         429  GOGETST1 CMP   #':'         ;Condition status flags
9EEE: 4C 20 D8      430  GOGETST2 JMP   GETSTT+3     ;Execute statement (token)
                    431
                    432  * Process token:
                    433
                    434  *  -> Check token using (gasp!) self-modifying code:
                    435
9EF1: 85 33         436  CHKTKN   STA   PROMPT      ;Store token in PROMPT and
9EF3: A8            437          TAY                ; use as lookup index to
9EF4: B9 99 B7      438          LDA   APTKNTBL-$80,Y ; Applesoft Token Table
9EF7: 30 F5         439          BMI   GOGETST2     ;Ordinary token so execute
9EF9: 8D FD 9E      440          STA   :1+1         ;Special token. Set handler
9EFC: D0 00         441  :1       BNE   BPRINT       ;Branch set by prior line
                    442
                    443  *  -> PRINT or IF token:
                    444
                    445  BIF
9EFE: 85 33         446  BPRINT   STA   PROMPT      ;PROMPT=0
9F00: 8D 4C BE      447          STA   CHRLAST      ;Wipe out last char
9F03: AD 03 B8      448          LDA   STATED       ;Set STATE D to check for
9F06: 8D 38 BE      449          STA   VSYSIO       ; CTL-D after PRINT token
9F09: AD 04 B8      450          LDA   STATED+1
9F0C: 8D 39 BE      451          STA   VSYSIO+1
9F0F: D0 1D         452          BNE   GOGETST3     ;Always
                    453
                    454  *  -> LIST token:
                    455
9F11: A9 01         456  BLIST    LDA   #1
9F13: 85 33         457          STA   PROMPT      ;PROMPT=1
9F15: D0 17         458          BNE   GOGETST3     ;Always
                    459
                    460  *  -> CALL token:
                    461
9F17: A9 02         462  BCALL    LDA   #2
9F19: 85 33         463          STA   PROMPT      ;PROMPT=2
9F1B: D0 11         464          BNE   GOGETST3     ;Always
                    465
                    466  *  -> LET token (forces check for garbage collection):
                    467
9F1D: 98            468  BLET     TYA               ;Put token in A
9F1E: 4C CE 9E      469          JMP   TRCFRE
                    470
                    471  *  -> TRACE token:
                    472
9F21: 8C 41 BE      473  BTRACE   STY   DTRACE      ;Set BI trace flag
9F24: D0 04         474          BNE   SETASTRC     ;Always
                    475
                    476  *  -> NOTRACE token:
                    477
9F26: 8D 41 BE      478  BNOTRACE STA   DTRACE      ;Clear BI trace flag
9F29: 88            479          DEY                ;Change NOTRACE to TRACE
9F2A: A9 4A         480  SETASTRC LDA   #$4A        ;Set Apsoft trace flag (TRACE
9F2C: 85 F2         481          STA   TRCFLG       ; handler will ROR to $A5)
```

continued on page 126

Gun For Hire

Black Sun's software rental library will blow you away!

Announcing an end to hit or miss software decisions

Selecting the right software can be pretty difficult. To say the least. Newer, better software is released every day. And 'state-of-the-art' is redefined every week or so. The solution? Renting. Black Sun's rental option makes computing on your Apple more enjoyable than ever. Without shooting your budget full of holes.

Over 2,000 rental titles for the Apple and Macintosh computers

Black Sun handles only one line of software. Apple. With years of experience you can rely on. A capable support staff maintains an up-to-date product library of all the latest releases and a well organized product catalog makes finding the right software a snap. Featured categories include adventure, business, communications, education, game, graphics, home/hobby/utility, strategy, word processing, hardware, and books. Plus hundreds of rare and ½ price items are made available through their unique consignment option.
A Black Sun exclusive.

Don't take a shot in the dark...rent it first!

You can rent software for 3 weeks for just **15 to 20%** of the list price, or less. All rental fees are applied to the competitive **discount** prices...if you decide to buy. You call the shots. Evaluate a business package, solve an adventure, try something new, or whatever! The choice is yours. As it should be.

Give it a shot!

Take advantage of this introductory offer and save $5.00 off the regular one-time membership fee of $25.00. There are **no** annual dues, hidden costs, or complicated deposit schemes. Send check or VISA/MasterCard info for $20.00 along with your name, address, phone #, and system info. Your product catalog and membership info will be sent immediately.

If you are not 100% satisfied with the service and/or selections, simply inform them within 30 days and your $20.00 will be promptly refunded!

Name: _____

Address: _____

City: _____ State _____ Zip: _____

Circle System: II+ //e //c Other Mac _____K

Or charge it to ☐ Visa VISA ☐ MC MasterCard

_____ Exp: _____

Signature _____ Date: _____

For more information call or write:

Black Sun
P.O. Box 406
Emmaus, PA 18049
(215) 967-3300

processed by the BI will the trace anomaly be corrected.

RIGHTING THE WRONGS

Having identified the trace bug, I set out to exterminate it. I turned off my //e, donned my thinking cap, and stretched out in my usual horizontal working position with a pad and pencil. During the composition of this article, I had a gut feeling that the trace character image (**line 521** of **Listing 1**) was unnecessary. After all, if we prove that STATE 4 (or STATE 0) flow comes from OUTRACE, the character in the Accumulator must be either the trace character proper ($A3) or, in FLASH mode, the pound sign ORed with $40 ($E3). This being the case, why test for the trace character at all? Instead, why not simply test for OUTRACE activity? If found, program flow passes to BITRACE1 (**line 383** of **Listing 1**); if not found, it continues processing through VSTATE4.

Listing 3 patches the beginning of VSTATE4 to do just that. **Lines 23-29** install the patch when FLASH.PATCH is executed. The actual patch starts by saving the entry Accumulator contents on the stack (**line 37**) and the entry X in the global page (**line 38**). As in the original code, a check is now made for OUTRACE output (**lines 39-45**). Note that we must search one byte deeper into the stack because of the PHA instruction in **line 37**. If output does not originate at OUTRACE (**lines 42 and 45**), flow branches to **line 48**, where the entry Accumulator is restored and three NOP bytes are used as fillers. Flow drops to **line 293** of **Listing 1**, where the entry X-Register is restored and STATE 4 handling continues. OUTRACE activity is routed to BITRACE1 via **lines 46-47**.

To install the patch temporarily, use your assembler to enter **Listing 3** in its entirety. If you don't have an assembler, enter the Monitor with CALL −151. Then key in the code as one continuous block starting at $8000, ignoring the second origin at **line 35** and its resulting change in addresses. Save the program with the command:

BSAVE FLASH.PATCH,A$8000,L$28

Installing the patch is then a simple matter of entering the command:

−FLASH.PATCH

To make the fix permanent, type:

BLOAD BASIC.SYSTEM,TSYS,A$2000

Enter the Monitor (CALL −151), type the actual patch (**lines 37-49** of **Listing 3**) into locations $282C-$2845, and type:

BSAVE BASIC.SYSTEM,TSYS,A$2000, L$2800

Note that the entry addresses and the final addresses in the PRODOS system file are different, although the low bytes correspond. If you are installing this patch in BI Version 1.0, start entering the code at

LISTING 1: BI Output Processing *(continued)*

```
9F2E: 98          482 GOGETST3 TYA              ;Put token in A
9F2F: 4C 20 D8    483          JMP   GETSTT+3   ;Execute Apsoft statement
                  484
                  485 *  -> RESUME token:
                  486
9F32: A9 00       487 BRESUME  LDA   #0
9F34: 85 DE       488          STA   ERRNUM     ;Clear Apsoft error code
9F36: 98          489          TYA              ;Put token in A
9F37: D0 B3       490          BNE   GOGETST1   ;Always
                  491
                  492 *  -> Do the tracing:
                  493
9F39: 20 00 9A    494 DOTRACE  JSR   SETIOTRU   ;Set true I/O handlers
9F3C: A9 A3       495          LDA   #"#"
9F3E: 20 ED FD    496          JSR   COUT       ;Print trace char
9F41: A6 75       497          LDX   CURLIN
9F43: A5 76       498          LDA   CURLIN+1
9F45: 20 24 ED    499          JSR   LINPRT     ;Print line number
9F48: A9 A0       500          LDA   #space
9F4A: 20 ED FD    501          JSR   COUT       ;Print space
9F4D: 20 8D 9A    502          JSR   SAVIOTRU   ;Restore intercepts
9F50: 38          503          SEC              ;For free space calc
9F51: 4C C6 9E    504          JMP   NXTASCHR
                  505
                  506 *  -> FLASH token:
                  507
9F54: A9 E3       508 BFLASH   LDA   #"#"+$40
9F56: D0 02       509          BNE   FLS1       ;Always
                  510
                  511 *  -> INVERSE or NORMAL token:
                  512
9F58: A9 A3       513 BNOFLS   LDA   #"#"       ;Apsoft trace char
9F5A: 8D 61 9F    514 FLS1     STA   TRACECHR
9F5D: 98          515          TYA              ;Put token in A
9F5E: D0 8C       516          BNE   GOGETST1   ;Always
9F60: 00          517          BRK              ;Never
                  518 *--------------------------------
                  519 * Trace Character Storage [$9F98]:
                  520 *--------------------------------
9F61: A3          521 TRACECHR ASC   "#"        ;Trace char
                  522 *--------------------------------
                  523 * Write Buffered Data [$A025]:
                  524 *--------------------------------
                  525          ORG   $9FEE
                  526
                  527 * Set for maximum of 256 bytes:
                  528
9FEE: A9 00       529 WRBUFDAT LDA   #0
9FF0: A0 01       530          LDY   #1         ;$100 bytes to write
9FF2: D0 13       531          BNE   WRBUFDA3   ;Always
                  532
                  533 * Write buffered data and check error:
                  534
9FF4: 20 00 A0    535 WRBUFDA1 JSR   WRBUFDA2   ;Write data
9FF7: 90 23       536          BCC   RTS6       ;No error. Exit
9FF9: A8          537          TAY              ;Error. Preserve A
9FFA: 68          538          PLA              ;Remove return address
9FFB: 68          539          PLA              ; from stack
9FFC: 98          540          TYA              ;Restore A
9FFD: 4C EE 9A    541          JMP   ERROR      ;Go to error handler
                  542
                  543 * Write buffered data to disk:
                  544
A000: A0 00       545 WRBUFDA2 LDY   #0
A002: AD 4A BE    546          LDA   TBUFPTR    ;Get data count (<256)
A005: F0 14       547          BEQ   CLCRTS1    ;No data found
A007: 8D D9 BE    548 WRBUFDA3 STA   RWCOUNT    ;Set # bytes to write
A00A: 8C DA BE    549          STY   RWCOUNT+1  ; in WRITE parmlist
A00D: A9 CB       550          LDA   #WRITE
A00F: 20 70 BE    551          JSR   GOSYSTEM   ;WRITE call
A012: 48          552          PHA              ;Save error code
A013: A9 00       553          LDA   #0
A015: 8D 4A BE    554          STA   TBUFPTR    ;Zero buffered data count
A018: 68          555          PLA              ;Restore error code
A019: B0 01       556          BCS   RTS6       ;CS=error
A01B: 18          557 CLCRTS1  CLC              ;CC=no error
A01C: 60          558 RTS6     RTS
                  559 *--------------------------------
                  560 * Applesoft Token Table [$B873]:
                  561 *--------------------------------
                  562          ORG   $B819
                  563
                  564 * Tokens handled normally (lower case) have hi
                  565 * bit set. Tokens requiring special handling
                  566 * (upper case) are replaced by values which
                  567 * are branch offsets in BI Trace Processing.
                  568
B819: 80 81 82    569 APTKNTBL HEX   808182     ;end, for, next
B81C: 83 84 85    570          HEX   838485     ;data, input, del
B81F: 86 87 88    571          HEX   868788     ;dim, read, gr
B822: 89 8A 8B    572          HEX   898A8B     ;text, pr#, in#
B825: 19 8D 8E    573          HEX   198D8E     ;CALL, plot hlin
B828: 8F 90 91    574          HEX   8F9091     ;vlin, hgr2, hgr
B82B: 92 93 94    575          HEX   929394     ;hcolor=, hplot, draw
B82E: 95 96 97    576          HEX   959697     ;xdraw, htab, home
B831: 98 99 9A    577          HEX   98999A     ;rot=, scale=, shload
```

```
B834: 23 28 5A  578           HEX   23285A      ;TRACE, NOTRACE, NORMAL
B837: 5A 56 A0  579           HEX   5A56A0      ;INVERSE, FLASH, color=
B83A: A1 A2 A3  580           HEX   A1A2A3      ;pop, vtab, himem:
B83D: A4 A5 34  581           HEX   A4A534      ;lomem:, onerr, RESUME
B840: A7 A8 A9  582           HEX   A7A8A9      ;recall, store, speed=
B843: 1F AB AC  583           HEX   1FABAC      ;LET, goto, run
B846: 00 AE AF  584           HEX   00AEAF      ;IF, restore, &
B849: B0 B1 B2  585           HEX   B0B1B2      ;gosub, return, rem
B84C: B3 B4 B5  586           HEX   B3B4B5      ;stop, on, wait
B84F: B6 B7 B8  587           HEX   B6B7B8      ;load, save def
B852: B9 00 BB  588           HEX   B900BB      ;poke, PRINT, cont
B855: 13 BD BE  589           HEX   13BDBE      ;LIST, clear, get
B858: BF        590           HEX   BF          ;new

--End assembly, 697 bytes, Errors: 0
```

END OF LISTING 1

LISTING 2: Applesoft ROM Routines
Note: Don't enter this code. It already exists in Applesoft ROM.

```
                1   ****************************************
                2   *                                      *
                3   *          Applesoft ROM Routines      *
                4   *                                      *
                5   *      Interpreted by Sandy Mossberg   *
                6   *                                      *
                7   ****************************************
                8
                9   * Merlin-Pro Assembler
                10
                11  INVFLG   =     $32         ;Character output mask
                12  CURLIN   =     $75         ;Current line#
                13  OLDTEXT  =     $79         ;Save TXTPTR
                14  FAC      =     $9D         ;Floating point accumulator
                15  CHRGET   =     $B1         ;Bump TXTPTR and get char
                16  CHRGOT   =     $B7         ;Get char at TXTPTR
                17  TXTPTR   =     $B8         ;Text pointer
                18  SPDBYT   =     $F1         ;SPEED value
                19  TRCFLG   =     $F2         ;Trace flag
                20  ORMASK   =     $F3         ;FLASH mask
                21  REMSTK   =     $F8         ;Save stack pointer
                22  CMDTBL   =     $D000       ;Statement command table
                23  RTN1     =     $D857       ;RTS instruction
                24  ISCNTC   =     $D858       ;Check for CTL-C
                25  GOCMDLP  =     $D88A       ;Go to CMDLP if carry clear
                26  GOTOCMD  =     $D93E       ;GOTO command
                27  ADDON    =     $D998       ;Add (Y) to TXTPTR
                28  REMN     =     $D9A6       ;Find EOL
                29  FRMEVL   =     $D9C9       ;Evaluate formula at TXTPTR
                30  LETCMD   =     $DA46       ;LET statement
                31  OUTSP    =     $DB57       ;Output space
                32  SYNCHR   =     $DEC0       ;Assure TXTPTR = (A)
                33  SYNERR   =     $DEC9       ;Print ?SYNTAX ERROR
                34  LINPRT   =     $ED24       ;Print decimal of (A,X)
                35  WAIT     =     $FCA8       ;Delay
                36  COUT     =     $FDED       ;Output char
                37
                38  goto     =     $AB         ;GOTO token
                39  then     =     $C4         ;THEN token
                40
                41  *-------------------------------
                42  * New Statement Handler:
                43  *-------------------------------
                44          ORG   $D7D2
                45
D7D2: BA        46  NEWSTT  TSX               ;Save stackpointer
D7D3: 86 F8     47          STX   REMSTK
D7D5: 20 58 D8  48          JSR   ISCNTC      ;Check CTL-C abortion
D7D8: A5 B8     49          LDA   TXTPTR      ;Prepare to save TXTPTR
D7DA: A4 B9     50          LDY   TXTPTR+1
D7DC: A6 76     51          LDX   CURLIN+1    ;Immediate mode ($FF in
D7DE: E8        52          INX               ;  CURLIN hi)?
D7DF: F0 04     53          BEQ   CKEND       ;Yes
D7E1: 85 79     54          STA   OLDTEXT     ;No. Save TXTPTR
D7E3: 84 7A     55          STY   OLDTEXT+1
D7E5: A0 00     56  CKEND   LDY   #0
D7E7: B1 B8     57          LDA   (TXTPTR),Y  ;Check byte at TXTPTR
D7E9: D0 57     58          BNE   EOS         ;Nonzero. Should be EOS
D7EB: A0 02     59          LDY   #2          ;Zero. Check absolute or
D7ED: B1 B8     60          LDA   (TXTPTR),Y  ;  relative link byte hi
D7EF: 18        61          CLC
D7F0: F0 34     62          BEQ   EOP         ;Zero. End program
D7F2: C8        63          INY               ;Nonzero. Deferred mode only
D7F3: B1 B8     64          LDA   (TXTPTR),Y
D7F5: 85 75     65          STA   CURLIN      ;Line# lo
D7F7: C8        66          INY
D7F8: B1 B8     67          LDA   (TXTPTR),Y
D7FA: 85 76     68          STA   CURLIN+1    ;Line# hi
D7FC: 98        69          TYA
D7FD: 65 B8     70          ADC   TXTPTR      ;Set TXTPTR to 1st char of
D7FF: 85 B8     71          STA   TXTPTR      ;  line contents
D801: 90 02     72          BCC   EXECUTE
```

continued on next page

$285B. Also, note that this patch is only good for BI Versions 1.0 and 1.1. It will not be compatible with subsequent releases of the BASIC interpreter. Presumably, Apple will have the bug fixed by then.

This generic solution to the trace character problem has widespread implications. All BI code employing TRACECHR may now be eliminated. Special processing of NORMAL, INVERSE and FLASH tokens also becomes superfluous. The net result is cleaner, less convoluted BI code.

The nonprocessing of tokens (e.g., RESUME, TRACE, NOTRACE) following true IF-THEN statements will still be a problem. Although not within the scope of this article, a solution may be found by processing IF-THEN statements within the BI itself. It's not complicated.

Implementation of these ideas requires rewriting a bit of the BI code. When Apple Computer gets around to revising Version 1.1, I sincerely hope they take these ideas into consideration. If you detect a flaw in my reasoning, please embarrass me.

UPCOMING

The next installment of Disassembly Lines will look at validation and processing of BI commands. Subsequently, we shall begin dissecting individual commands.

LISTING 2: Applesoft ROM Routines *(continued)*

```
D803: E6 B9      73              INC   TXTPTR+1
                 74
D805: 24 F2      75   EXECUTE    BIT   TRCFLG      ;Trace active?
D807: 10 14      76              BPL   GETSTT      ;No
D809: A6 76      77              LDX   CURLIN+1    ;Yes.
D80B: E8         78              INX
D80C: F0 0F      79              BEQ   GETSTT      ;Immediate mode. Skip trace
D80E: A9 23      80              LDA   #'#'        ;Deferred mode.
D810: 20 57 DB   81   OUTRACE    JSR   OUTDO       ;Print "#" (BI TRAPS THIS!)
D813: A6 75      82              LDX   CURLIN
D815: A5 76      83              LDA   CURLIN+1
D817: 20 24 ED   84              JSR   LINPRT      ;Print decimal line#
D81A: 20 57 DB   85              JSR   OUTSP
D81D: 20 B1 00   86   GETSTT     JSR   CHRGET      ;Get next char
D820: 20 28 D8   87              JSR   TYPSTT      ;Evaluate and process it
D823: 4C D2 D7   88              JMP   NEWSTT      ;Back for more
                 89
D826: F0 62      90   EOP        BEQ   GOCMDLP     ;Route flow to CMDLP
                 91
D828: F0 2D      92   TYPSTT     BEQ   RTN1        ;Allow leading/trailing colon
D82A: E9 80      93              SBC   #$80        ;Token or nontoken?
D82C: 90 11      94              BCC   LETSTT      ;Nontoken. Must be LET statement
D82E: C9 40      95              CMP   #$40        ;Token
D830: B0 14      96              BCS   GOSYNERR    ;Syntax error if token > $BF
D832: 0A         97              ASL               ;Double index to CMDTBL
D833: A8         98              TAY               ;Transfer to (Y)
D834: B9 01 D0   99              LDA   CMDTBL+1,Y
D837: 48         100             PHA               ;Hi byte on stack
D838: B9 00 D0   101             LDA   CMDTBL,Y
D83B: 48         102             PHA               ;Lo byte on stack
D83C: 4C B1 00   103             JMP   CHRGET      ;Get next char then RTS-jump
                 104
D83F: 4C 46 DA   105   LETSTT     JMP   LETCMD      ;Execute LET (assign) statement
                 106
D842: C9 3A      107   EOS        CMP   #':'        ;Colon?
D844: F0 BF      108             BEQ   EXECUTE     ;Yes. Process next statement
D846: 4C C9 DE   109   GOSYNERR   JMP   SYNERR      ;No. Syntax error
                 110   *--------------------------------
                 111   * Output Character:
                 112   *--------------------------------
                 113             ORG   $DB57
                 114
DB57: 09 80      115   OUTDO      ORA   #$80        ;Assure negative ASCII
DB59: C9 A0      116              CMP   #$A0        ;Control char?
DB5B: 90 02      117              BCC   PRCHR       ;Yes. Print invisible char
DB5D: 05 F3      118              ORA   ORMASK      ;FLASH=$40. NORMAL/INVERSE=0
DB5F: 20 ED FD   119   PRCHR      JSR   COUT        ;Output char
DB62: 29 7F      120              AND   #$7F        ;Convert to positive ASCII
DB64: 48         121              PHA               ;Save on stack
DB65: A5 F1      122              LDA   SPDBYT      ;Fastest=1. Slowest=$FF
DB67: 20 A8 FC   123              JSR   WAIT        ;Delay
DB6A: 68         124              PLA               ;Restore original char
DB6B: 60         125              RTS               ;Back to caller
                 126   *--------------------------------
                 127   * IF Statement Handler:
                 128   *--------------------------------
                 129             ORG   $D9C9
                 130
D9C9: 20 C9 D9   131   IFCMD      JSR   FRMEVL      ;Evaluate formula after IF
D9CC: 20 B7 00   132              JSR   CHRGOT      ;Get char after formula
D9CF: C9 AB      133              CMP   #goto
D9D1: F0 05      134              BEQ   :1          ;GOTO found
D9D3: A9 C4      135              LDA   #then
D9D5: 20 C0 DE   136              JSR   SYNCHR      ;Assure THEN found
D9D8: A5 9D      137   :1         LDA   FAC
D9DA: D0 05      138              BNE   IFTRU       ;IF evaluates true
                 139   *--------------------------------
                 140   * REM (or False IF) Statement Handler:
                 141   *--------------------------------
D9DC: 20 A6 D9   142   REMCMD     JSR   REMN        ;Skip remainder of line
D9DF: F0 B7      143              BEQ   ADDON       ;Always
                 144   *--------------------------------
                 145   * True IF Statement Handler:
                 146   *--------------------------------
D9E1: 20 B7 00   147   IFTRU      JSR   CHRGOT      ;Get char after GOTO/THEN
D9E4: B0 03      148              BCS   :1          ;Command (statement) found
D9E6: 4C 3E D9   149              JMP   GOTOCMD     ;Number found
D9E9: 4C 28 D8   150   :1         JMP   TYPSTT      ;Execute command
                 151   *--------------------------------
                 152   * TRACE Statement Handler:
                 153   *--------------------------------
                 154             ORG   $F26D
                 155
F26D: 38         156   TRACECMD   SEC
F26E: 90         157              HEX   90          ;Skip next byte
                 158   *--------------------------------
                 159   * NOTRACE Statement Handler:
                 160   *--------------------------------
F26F: 18         161   NOTRCCMD   CLC
F270: 66 F2      162              ROR   TRCFLG      ;Shift carry into trace flag
F272: 60         163              RTS
                 164   *--------------------------------
                 165   * NORMAL Statement Handler:
                 166   *--------------------------------
F273: A9 FF      167   NORMCMD    LDA   #-1         ;Clear output mask
F275: D0 02      168              BNE   NOFLASH     ;Always
```

```
                    169  *------------------------------
                    170  * INVERSE Statement Handler:
                    171  *------------------------------
F277: A9 3F         172  INVCMD   LDA   #$3F        ;Set inverse output mask
F279: A2 00         173  NOFLASH  LDX   #0         ;Clear flash mask
F27B: 85 32         174  SETMASK  STA   INVFLG     ;Set output mask
F27D: 86 F3         175          STX   ORMASK     ;Set flash mask
F27F: 60            176          RTS
                    177  *------------------------------
                    178  * FLASH Statement Handler:
                    179  *------------------------------
F280: A9 7F         180  FLASHCMD LDA   #$7F        ;Set flashing output mask
F282: A2 40         181          LDX   #$40        ;Set flash mask
F284: D0 F5         182          BNE   SETMASK     ;Always

--End assembly, 200 bytes, Errors: 0
```

END OF LISTING 2

LISTING 3: FLASH.PATCH
See instructions to enter.

```
                      1  ************************************
                      2  *          FLASH.PATCH             *
                      3  *                                  *
                      4  *  Corrects THEN-FLASH bug in BI   *
                      5  *  [BI 1.1 addresses bracketed]    *
                      6  *       by Sandy Mossberg          *
                      7  *       Copyright (C) 1986         *
                      8  *       by MicroSPARC, Inc.        *
                      9  *       Concord, MA  01742         *
                     10  ************************************
                     11
                     12  * Merlin-Pro Assembler
                     13
                     14  STACK    =     $100        ;Stack
                     15  BITRACE1 =     $9EB6       ;BI trace processing [$9EE1]
                     16  PREGX    =     $BE3F       ;Save X-reg [same]
                     17  OUTRACE  =     $D810       ;Trace char output
                     18
                     19           ORG   $8000
                     20
                     21  * Move PATCH to BI:
                     22
8000: A2 00          23           LDX   #0
8002: BD 0E 80       24  MV1      LDA   PATCH,X
8005: 9D 2C 9E       25           STA   VSTATE4,X
8008: E8             26           INX
8009: E0 1A          27           CPX   #PATCH1-PATCH
800B: 90 F5          28           BCC   MV1
800D: 60             29           RTS
                     30
                     31  * PATCH corrects THEN-FLASH bug:
                     32
                     33  PATCH    =     *           ;Starting load address of patch
                     34
                     35           ORG   $9E2C       ;[$9E5B]
                     36
9E2C: 48             37  VSTATE4  PHA               ;Save entry A
9E2D: 8E 3F BE       38           STX   PREGX       ;Save entry X
9E30: BA             39           TSX               ;Use stack pointer as offset
9E31: BD 04 01       40           LDA   STACK+4,X   ;Check stack for Apsoft
9E34: C9 12          41           CMP   #OUTRACE+2  ; trace printing
9E36: D0 0A          42           BNE   :1          ;Not trace printing
9E38: BD 05 01       43           LDA   STACK+5,X
9E3B: C9 D8          44           CMP   #>OUTRACE+2
9E3D: D0 03          45           BNE   :1          ;Not trace printing
9E3F: 68             46           PLA               ;Trace printing. Restore A
9E40: B0 74          47           BCS   BITRACE1    ;Always
9E42: 68             48  :1       PLA               ;Not trace printing. Restore A
9E43: EA EA EA       49           HEX   EAEAEA      ;Filler NOP bytes
                     50
                     51           ORG               ;Recall load address
                     52
                     53  PATCH1   =     *           ;Ending load address+1 of patch

--End assembly, 40 bytes, Errors: 0
```

END OF LISTING 3

NIBBLE LIGHT PEN

by David Gauger, II

HARDWARE CONSTRUCTION PROJECT

Ten dollars worth of

parts and a short machine language program are all you need to add this convenient accessory to your Apple. An Applesoft demonstration program illustrates techniques to use the light pen to select items from menus.

Most commercially available light pen systems use hardware to detect the location of the pen on the screen. This allows excellent resolution; some systems are even accurate to one pixel. They are also expensive, both in terms of hardware cost and system complexity, making them impractical for the average home user.

This article describes a light pen that is inexpensive and easy to construct. It uses just three active electronic parts that cost about $10 and an ordinary ballpoint pen case. Despite its simplicity, the Nibble Light Pen is reliable and provides an effective, direct way to interact with any of the Apple // family computers. Like the mouse, the Nibble Light Pen lets you bypass the keyboard. Just point the light pen at the screen — you don't even need to "click."

The Nibble Light Pen consists of two basic elements: the light-sensing hardware and a machine language driver routine. Briefly, when the light pen detects the light from an inverse block on the screen, it transmits a pulse to one of the lines in the Apple // game port. This signals the driver routine to determine the screen position and store the coordinates in memory. The calling program can then PEEK these memory locations to get the coordinates.

David Gauger, II, 1430 E. 55th Pl., Tulsa, OK, 74105. Nibble Light Pen is compatible with DOS 3.3 and ProDOS.

THE HARDWARE

Some hardware-based light pen systems use the time it takes for the monitor's electron beam to get from the bottom of the screen to the top to calculate the pen's location. Others interrupt the microprocessor when the pen detects the raster scan on a certain line. To keep the hardware simple, this system uses software to determine the pen's position. The hardware's sole function is to detect light.

The light pen is equipped with a photo-Darlington light detector. This is a sensitive but fairly inexpensive semiconductor that interfaces to the Apple // game port with the help of resistors.

There are three pushbutton inputs in the Apple // game port. Each one corresponds to a specific address in memory, and to a specific pin in the game port connector. When the light pen is pointed at a single inverse block, the photo-Darlington delivers about +1 volt to a pushbutton input. If the resulting voltage on that button's pin is +1 volt or more, a value of 128 (hex $80) or greater is stored in the corresponding memory address. To see if the button has been "pushed" by the light pen, the pen software reads the location (address) of the button. This is accomplished in BASIC by a PEEK statement.

Although it will work adequately without it, I added a variable resistor to the light pen. This component makes it possible to vary the light threshold level at which the photo-Darlington delivers +1 volt to the pushbutton input. It increases the circuit's sensitivity, allowing the pen to respond adequately to much dimmer light levels.

THE DRIVER

The software driver (**Listing 1**) is a short machine language routine that resides in memory page 3. It is designed to be used as a subroutine that can be CALLed by a machine language or BASIC program.

When CALLed, the driver searches every point on the screen for an inverse block. When it finds one, it determines whether this is the block at which the pen is pointing. If not, the routine looks for the next block. When the driver finds the correct block, it sounds a tone to notify the user, stores the vertical coordinate of the block in location 768 (hex $300), stores the horizontal coordinate in location 769 (hex $301), and returns to BASIC. From BASIC your main program can PEEK these two locations to find out where the pen is pointing.

CONSTRUCTION

To get the system up and running, con-

FIGURE 1: Schematic for Apple II, II Plus, and //e Light Pen (DIP Connector)
(Inset: Lead Diagram for Photo-Darlington [Sylvania ECG 3036 or Motorola MRD 360])

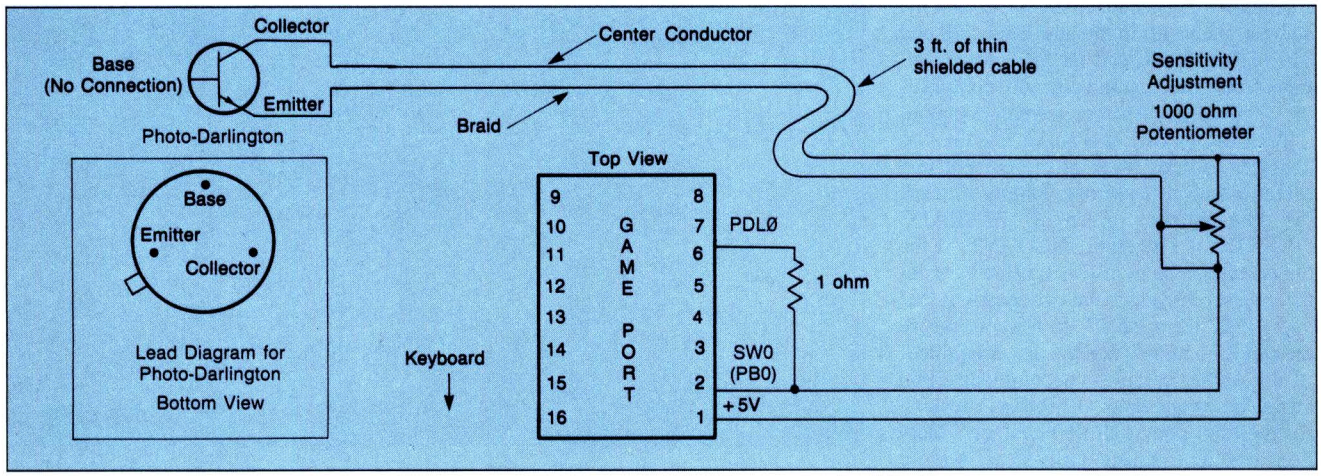
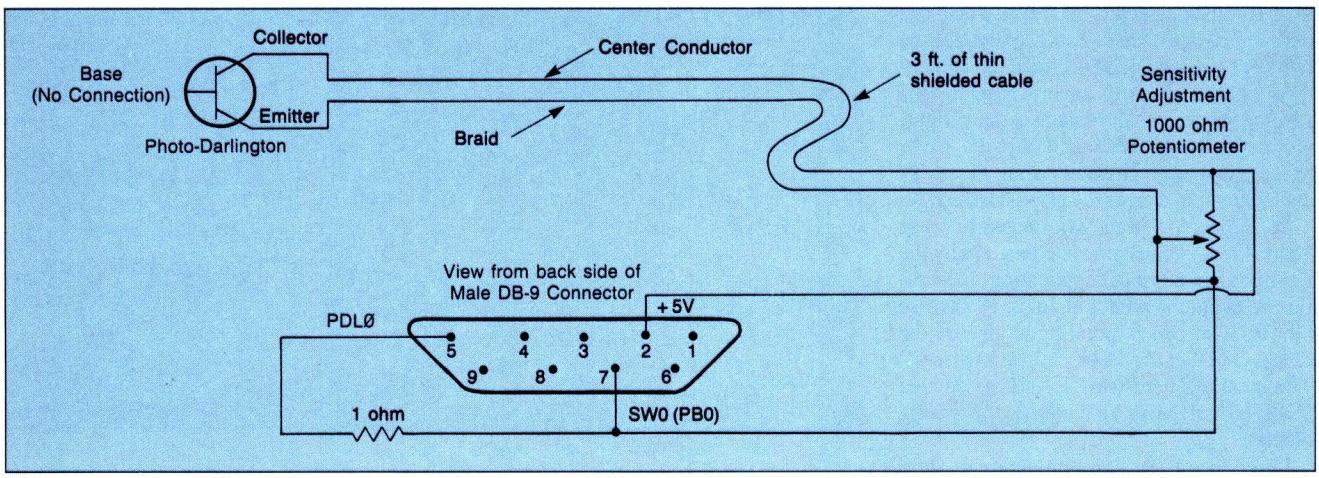


FIGURE 2: Schematic for Apple //c and //e Light Pen (DB Connector)



struct the light pen as shown in the diagram appropriate to your machine. If your computer uses a 16-pin DIP game socket (Apple II, II Plus or //e), then use the parts listed in **Table 1**, following the schematic diagram in **Figure 1**. Use **Figure 2** and the parts in **Table 2** if you have a //c or have equipped your machine with a DB-9 game socket. Next, type in the assembly listing (**Listing 1**), which is the same for all machines. If you do not have an assembler, consult the instructions in "A Welcome to New *Nibble* Readers" for help. Save the program to disk with the statement:

BSAVE PEN.DRIVER, A$302,L$8C

Any shielded cable may be used, but the more flexible it is, the easier the pen will be to manipulate. I used a disposable type of pen barrel to house the photo-Darlington, but anything that resembles a pen will do. Just be sure to feed the shielded cable down the center before you solder the light detector in place. Heat the leads of the photo-Darlington as little as possible, and mount it to provide close proximity to the screen (1/8 inch or less).

Any photo-Darlington may be used, provided it has adequate sensitivity for this application. You may have difficulty finding one though, since photo-Darlingtons are used almost exclusively in industrial applications, and electronics stores like Radio Shack do not carry them. An industrial electronics supply house should do the trick.

ADJUSTING THE LIGHT PEN'S SENSITIVITY

When the pen is constructed and the software has been loaded, use the following procedure to adjust the light pen's sensitivity. Set the brightness and contrast of the monitor to your liking. With the driver in memory, type in and run this short test program:

```
10 HOME
20 VTAB 10: HTAB 12
30 INVERSE
40 PRINT " ": NORMAL: REM 1 SPACE
50 CALL 770 : REM LIGHT PEN DRIVER
60 PRINT PEEK(768)
70 PRINT PEEK(769)
```

The Apple responds by placing an inverse space on the screen. Note that this square of light is probably blinking or flickering. If it is not, try turning the variable resistor to one end or the other of its range. If you still cannot get it to flicker, there is probably something wrong with the light pen or the driver software.

Turn the sensitivity adjustment to the end of its range so that the light blinks on and off very rapidly. The sensitivity is now at its lowest level. Turn the adjustment up until the light begins to flicker, then back it off slightly. The sensitivity is now at its highest, and the pen will respond to lower light levels from the screen. Touch the pen to the inverse space. The Apple should beep and then print a 9 above an 11. These are the coordinates of the block the driver found.

Set the sensitivity so that the light pen system locates the correct block every time. There should be a range of settings that make this possible. If the setting is too high or too low, the driver will sometimes return with the wrong block. The light pen system is now finished and ready to be used.

THE SOFTWARE

Because of the disjointed arrangement of text screen memory in the Apple, it is difficult to keep track of screen positions with simple row and column pointers. Fortunately, there is a built-in ROM routine called BASCALC that calculates the starting address of a screen row, given the number of

the row. Just place the row number in the Accumulator, execute BASCALC, and the address shows up in the addresses $28 and $29 (called BASL and BASH). An offset, representing the column position, completes the calculation of a screen memory location.

With BASCALC, it is possible to use two counters to keep track of the current text screen position. I use one counter to track the line number given to BASCALC (VCOUNT). The other, HCOUNT, is the offset value, which, in effect, is the horizontal counter. These are the two locations that you PEEK from BASIC to find out where the pen is pointing when the driver is called.

The Apple screen can accommodate 960 characters at one time. Obviously, checking for the light pen at all 960 character locations would be unnecessarily slow. A more efficient method would be to scan the screen for a pre-selected character. When scanning for a specific character, the driver would test perhaps 10 or 20 character locations for the light pen, instead of 960. Since the pen detects light, I chose to have it scan for the inverse space, which radiates a lot of light. However, it is possible to modify the driver to scan for any character that gives off enough light to trigger the photo-Darlington.

We've narrowed the scan down to 10 or 20 locations. Which one is pointed to by the light pen? In a typical application, the routine scans the screen and finds 10 inverse blocks. The pen is pointing to one of the blocks, but 9 of the 10 blocks are "wrong." The most efficient way to find the correct block would be to first detect the wrong blocks.

I used the process of elimination to identify the correct block. As long as the light pen points at an inverse square, it transmits +1 volt to the pushbutton input (PB0). To test a block, the routine replaces the inverse space with a regular, dark space, and it checks the voltage at the PB0. If the PB0 still holds +1 volt, the pen is not pointing at the darkened test block.

The "block off" test alone is not sufficient to positively identify the correct block. For instance, suppose the pen is pointing at the ground when the driver is called. In this case, since the pen detects no light, it does not transmit voltage to the PB0. However, when the driver tests the first block, it assumes that the light pen is pointing at the darkened test block. The solution is to add a second test: turn the test block back on by storing an inverse space there again, and check the PB0 to see if the light pen again transmits +1 volt.

These two tests usually locate the correct block, but they're not foolproof. To increase accuracy, I added another block off test. With adequate brightness and the sensitivity level set correctly, the reliability of these three tests approaches 100%.

When it does find the correct block, the driver replaces the space with an inverse block (the last test is a block off test) and sounds a two-pitch bell different from the Apple's bell. The horizontal and vertical counters already indicate the correct coordinates, so the driver returns to the calling program with an RTS.

If the driver does not find the correct block during the first scan down the screen, it returns to the top and scans again. The routine will return to the calling program only if it can find the correct block.

TABLE 1: Parts List for the Apple II, II Plus and //e DIP Connector

Item	Quantity	Source	Cost
Photo-Darlington	1	Industrial electronics supplier Sylvania #ECG 3036 or RCA #MRD 360	$4.75
1 kilohm potentiometer	1	Radio Shack (#271-227)	$0.59
1 ohm resistor	1	Industrial electronics supplier (Radio Shack does not stock)	$0.90
16-pin DIP connector	1	Radio Shack (#276-1980)	$1.69
Shielded cable	3 ft.	Radio Shack (#278-1277)	$2.39
Disposable pen (BIC Biro works well)	1		

TABLE 2: Parts List for the Apple //c and //e DB-9 Game Ports

Item	Quantity	Source	Cost
Photo-Darlington	1	Industrial electronics supplier Sylvania #ECG 3036 or RCA #MRD 360	$4.75
1 kilohm potentiometer	1	Radio Shack (#271-227)	$0.59
1 ohm resistor	1	Industrial electronics supplier (Radio Shack does not stock)	$0.90
Male DB-9 connector	1	Radio Shack (#276-1537)	$1.99
Shielded cable	3 ft.	Radio Shack (#278-1277)	$2.39
Disposable pen (BIC Biro works well)	1		

INSTALLING THE SYSTEM

To incorporate the Nibble Light Pen system into your own program, first place an inverse block or blocks on the screen. In BASIC, the code might look like this:

```
10 VTAB 10: HTAB 12: INVERSE:
   PRINT " ": NORMAL
```

Next, CALL 770 (hex $302) which is the driver, and touch the light pen to any inverse block on the screen. When the software has found the spot you're pointing at (it only takes a fraction of a second), you will hear a beep. Locations 768 and 769 now contain the vertical and horizontal coordinates of the inverse block you indicated.

It is important to note that the horizontal and vertical counters start counting at zero.

LISTING 1: PEN.DRIVER

```
1     ************************
2     *      PEN.DRIVER      *
3     * BY DAVID GAUGER II  *
4     * COPYRIGHT (C) 1986  *
5     * BY MICROSPARC, INC  *
6     * CONCORD, MA  01742  *
7     ************************
8     *
9     * MERLIN ASSEMBLER
10
```

```
                        11
                        12   * THIS PROGRAM EXPECTS A PHOTO-DARLINGTON TO BE CONNECTED
                        13   * FROM PIN 1 TO PIN 2 OF THE GAME PORT.
                        14   * INPUT IS READ FROM PBØ (PIN 2): ADDRESS $CØ61 (-16287)
                        15   * ADDITIONALLY, A 1 OHM RESISTOR IS NEEDED FROM PIN 2
                        16   * TO PIN 6 OF THE GAME PORT.
                        17
                        18
                        19            ORG   $3Ø2
                        2Ø   PEN      EQU   $CØ61
                        21   VCOUNT   EQU   $3ØØ
                        22   HCOUNT   EQU   $3Ø1
                        23   BASCALC  EQU   $FBC1
                        24   BASL     EQU   $28
                        25   WAIT     EQU   $FCA8
                        26   SPEAKER  EQU   $CØ3Ø
                        27   BELL     EQU   $FBDD
                        28
Ø3Ø2: AØ ØØ             29   SETUP    LDY   #$ØØ
Ø3Ø4: 8C ØØ Ø3          3Ø            STY   VCOUNT
Ø3Ø7: A9 ØØ             31            LDA   #$ØØ
Ø3Ø9: 2Ø C1 FB          32            JSR   BASCALC     ;CALC ADDRESS OF FIRST LINE
                        33
                        34
Ø3ØC: B1 28             35   MAINLOOP LDA   (BASL),Y   ;GET CHARACTER
Ø3ØE: C9 2Ø             36            CMP   #$2Ø        ;IS IT A BLOCK?
Ø31Ø: FØ Ø8             37            BEQ   PENTEST     ;YES -- TEST PEN
Ø312: C8                38   NEXTSPOT INY               ;NO
Ø313: CØ 28             39            CPY   #$28        ;END OF LINE?
Ø315: FØ 2F             4Ø            BEQ   NEWLINE     ;YES
Ø317: 4C ØC Ø3          41            JMP   MAINLOOP    ;NO - DO IT ALL AGAIN
                        42
                        43
Ø31A: A9 AØ             44   PENTEST  LDA   #$AØ
Ø31C: 91 28             45   TEST1    STA   (BASL),Y   ;TURN OFF BLOCK
Ø31E: 2Ø 58 Ø3          46            JSR   DETECT      ;ANY LIGHT?
Ø321: 9Ø Ø7             47            BCC   TEST2       ;NO - TEST PASSES
Ø323: A9 2Ø             48            LDA   #$2Ø        ;YES - TEST FAILS
Ø325: 91 28             49            STA   (BASL),Y   ;WHERE YOU GOT IT
Ø327: 4C 12 Ø3          5Ø            JMP   NEXTSPOT    ;AND TRY NEXT SPOT
Ø32A: A9 2Ø             51   TEST2    LDA   #$2Ø
Ø32C: 91 28             52            STA   (BASL),Y   ;PUT BLOCK BACK
Ø32E: 2Ø 58 Ø3          53            JSR   DETECT      ;ANY LIGHT?
Ø331: BØ Ø3             54            BCS   TEST3       ;YES - TEST PASSES
Ø333: 4C 12 Ø3          55            JMP   NEXTSPOT    ;NO - TRY NEXT SPOT
Ø336: A9 AØ             56   TEST3    LDA   #$AØ
Ø338: 91 28             57            STA   (BASL),Y   ;TURN BLOCK BACK OFF
Ø33A: 2Ø 58 Ø3          58            JSR   DETECT      ;ANY LIGHT?
Ø33D: 9Ø 43             59            BCC   EXIT        ;NO - TEST PASSES - BLOCK FOUND
Ø33F: A9 2Ø             6Ø            LDA   #$2Ø
Ø341: 91 28             61            STA   (BASL),Y   ;SO PUT BLOCK BACK
Ø343: 4C 12 Ø3          62            JMP   NEXTSPOT    ;AND TRY AGAIN
                        63
Ø346: EE ØØ Ø3          64   NEWLINE  INC   VCOUNT     ;INCREMENT VERT COUNTER
Ø349: AD ØØ Ø3          65            LDA   VCOUNT
Ø34C: C9 18             66            CMP   #$18        ;IS IT MORE THAN THE 24TH LINE?
Ø34E: FØ B2             67            BEQ   SETUP       ;YES - TIME TO START AT THE TOP
Ø35Ø: 2Ø C1 FB          68            JSR   BASCALC     ;NO - FIGURE NEW BASE ADDRESS
Ø353: AØ ØØ             69            LDY   #$ØØ        ;ZERO HORIZONTAL COUNTER
Ø355: 4C ØC Ø3          7Ø            JMP   MAINLOOP    ;AND DO IT ALL AGAIN
                        71
Ø358: 8C Ø1 Ø3          72   DETECT   STY   HCOUNT     ;SAVE HORIZONTAL COUNTER
Ø35B: A2 Ø8             73            LDX   #$Ø8        ;SET UP COUNTER
Ø35D: AØ ØØ             74            LDY   #$ØØ        ;LOAD Y COUNTER
Ø35F: AD 61 CØ          75   GETPEN   LDA   PEN         ;SEE IF PEN SAW LIGHT
Ø362: 3Ø ØB             76            BMI   YESLITE
Ø364: 88                77            DEY
Ø365: DØ F8             78            BNE   GETPEN      ;THIS LOOP TESTS THE PEN
Ø367: CA                79            DEX               ;MANY TIMES
Ø368: DØ F5             8Ø            BNE   GETPEN
Ø36A: 18                81            CLC               ;CLEAR CARRY: NO LIGHT
Ø36B: AC Ø1 Ø3          82            LDY   HCOUNT      ;RESTORE HORIZONTAL COUNTER
Ø36E: 6Ø                83            RTS
Ø36F: 38                84   YESLITE  SEC               ;SET CARRY: YES LIGHT
Ø37Ø: AC Ø1 Ø3          85            LDY   HCOUNT      ;RESTORE HORIZONTAL COUNTER
Ø373: 6Ø                86            RTS
                        87
Ø374: AØ CØ             88   BELL2    LDY   #$CØ        ;LENGTH OF BELL2
Ø376: A9 Ø8             89   LOOP     LDA   #$Ø8        ;PITCH OF BELL2
Ø378: 2Ø A8 FC          9Ø            JSR   WAIT
Ø37B: AD 3Ø CØ          91            LDA   SPEAKER     ;CLICK SPEAKER
Ø37E: 88                92            DEY               ;DECREMENT COUNTER
Ø37F: DØ F5             93            BNE   LOOP        ;DO IT ALL AGAIN
Ø381: 6Ø                94            RTS
                        95
Ø382: 18                96   EXIT     CLC               ;FOR SAFETY
Ø383: A9 2Ø             97            LDA   #$2Ø
Ø385: 91 28             98            STA   (BASL),Y   ;WHERE YOU GOT IT
Ø387: 2Ø DD FB          99            JSR   BELL        ;REGULAR BELL
Ø38A: 2Ø 74 Ø3          1ØØ           JSR   BELL2       ;DIFFERENT BELL FOR FUN
Ø38D: 6Ø                1Ø1           RTS

--End assembly--

14Ø bytes

Errors: Ø
```

END OF LISTING 1

continued on page 136

The vertical counter ranges from 0-23, and the horizontal, from 0-39. If the driver returns with a vertical value of 16 and a horizontal value of 20, this is the same spot on the screen as one defined by VTAB 17 and HTAB 21.

The pen's resolution is partly determined by the photo-detector. I have used photo-Darlingtons that can resolve one block of light in two. This means that an inverse block cannot be placed immediately adjacent on any side. The photo-Darlington can adequately resolve light blocks immediately diagonal to one another fairly well. Should you want to place a block on every line or column, the blocks must be placed diagonally in zigzag fashion in order for the pen to discern them.

APPLICATIONS

Your imagination is the only limit to your applications of the Nibble Light Pen. For example, a typical instruction screen might have the prompt at the bottom: "Press <RETURN> to continue." Instead of using an INPUT or GET statement to control program flow at this point, place one inverse space on the screen labeled "Touch pen here to continue." Then call the light pen driver. The driver will not return to the calling program until it finds the block at which the pen is pointing. The result is effective program control without using the keyboard.

Another obvious use is in menu selection. In any computer magazine you'll find plenty of advice on menu input, how to organize menu screens, and the error trapping that inevitably accompanies keyboard input. Using a light pen with a menu eliminates many of these problems.

I have written a simple demonstration program to illustrate these two applications. Simply type in **Listing 2**, and save it on a disk that already contains the light pen driver (PEN.DRIVER, **Listing 1**) with the command:

SAVE LIGHT.PEN.DEMO

This demonstration program uses the light pen in the most simple and elementary ways; it's just meant to get you started. Other applications could include graphics, data input, screen layout and games. It may also be an ideal input device for people with certain handicaps, and it is flexible enough to be incorporated into just about any program or language.

MODIFICATIONS

Here are a few ideas for changes you can make. The routine can be changed to scan the screen for characters other than the inverse block. It could also be adapted to scan for a range of characters, such as the entire inverse character set or just inverse numbers. I have found the system sensitive enough to detect regular video characters, although the period and comma present some difficulty.

You may find that from time to time the driver makes a mistake and returns with the coordinates of a block to which you are not pointing. In writing the driver I sacrificed a bit of accuracy in favor of speed. The portion of code that actually detects light from the pen, the DETECT subroutine, is the culprit. Specifically, **lines 74 and 75** are counters that form loops to check the pen for light 2,048 times, each time the subroutine is called. Obviously, this takes a bit of time even at machine language speed, but the more times you check the pen for light, the more accurate your results will be. If you require more accuracy and are willing to forego execution speed, I suggest that you LDX with 0A in **line 74**. If you're assembling the driver at hex $302, this means that you store 0A at location hex $35C.

Resolution could perhaps be improved by changing the photo-Darlington, arranging a tube to narrow the area of light it responds to, or substituting another photo-detector such as a photo-diode or light-dependent resistor. Another idea is to use the low resolution graphics mode to scan for the pen. Theoretically, this should double the pen's resolution. Also, there is no reason why another pushbutton input could not be used. Using PB2 would allow you to use the paddles or a joystick simultaneously with the light pen (this is not feasible on the //c).

HARDWARE

SMARTMODEM 300 **$129**, HAYES MICROMODEM IIe with SMARTCOM-1 Software **$129**
1200-300 Baud External Modem.**$229**, Serial Cable **$20**, CP/M Z-80 Card. **$79**
64K 80 column Card IIe..**$89**, Video-7 RGB IIe Card.**$119**, Video-7 RGB I/F IIc...**$89**
64K 80 column IIe Display Enhancer Card with 2 outputs for Mono & RGB monitors**$129**
512K Ramworks IIe.$199, **1Mb Ramworks...$299**, RGB Module...**$109**
Grappler C Graphic Parallel Printer I/F **$89**, GRAPPLER+IIe/II+.**$89**, 64K Grappler**$139**
Interface and Cable for connecting APPLE IIc to any Centronics type Parallel Printer..**$69**
=> 32K Buffer for any Parallel Printer.$89, 128K.$99, 512K.$399
Excellent Quality **half ht. DRIVE** for **Apple II+,IIe,IIc**(Specify).**$129, Dual Drv$299**
KOALAPAD+**$79**,SYSTEM SAVER**$59**,**Dual Voltage 220/110V** Power Supply(IIe) **$79**
SWITCHBOX(Parallel/Serial) A/B..**$79**: A/B/C/D..**$99**: X switch(2 CPUs-2 printers)....**$99**
Cable for Mac or Apple IIc to Imagewriter-2 or -1 or Standalone Modem-Specify**$20** Ea.
6 OUTLET UL Listed Power Strip with Surge Suppressor & MCB and Indicating Light.**$19**
256K MEMORY SET(8 CHIPS) for RAMWORKS or MULTIRAM Card 150 nano-seconds.....**$29**
64K MEMORY SET(8 CHIPS) FOR PRINTER BUFFER CARDS, ETC 200 nano-seconds..... **$8**

FOR IMAGEWRITER COMPATIBLE 180cps PROWRITER, CITIZEN, EPSON PRINTERS, & HIGH-RES TAXAN MONITORS, PLEASE CALL FOR LOW PRICES.

RIBBONS FOR PRINTERS(CALL FOR OTHER RIBBONS):

CARTRIDGE for **IMAGEWRITER, NEC 8023, PROWRITER..$5 Ea. For Scribe.$7Ea.**
CARTRIDGE for **EPSON** MX-80/RX-80/FX-80/FX-85/**CITIZEN** 10 and **LX-80/90..$5 Ea.**
CARTRIDGE for **OKI-182/192..$8**; SPOOL for **OKIDATA**-82 or 92, GEMINI 10X.**$3Ea.**
Flip-N-File 15..**$6**, Flip File for 50 Diskettes.**$11.95**, Flip File For 100 Diskettes **$19.95**

Box of 10 Disks.(5.25in.)..$9 **SOFTWARE** **Disk Notcher(Use other Side)..$5**

NEW Prodos version of PFS File, Write & Plan for Apple IIe,c Hard drive compat. **$79ea.**
PFS File, Report, Write, Graph, Plan for IIe,IIc...**$79ea**, Magic Window 64K or 128K..**$99**
APPLEWORKS e/c & IIe Keyboard Template.$229, **Speller** for Appleworks **$99**
Quickfile IIe...**$89** : **Superpilot** with **Log IIe..$189**: LOGO for Apple IIe,IIc...**$99**
Webster's New World Spelling Checker for IIe,IIc,II+ (Checks 110,000+words).....**$35**
ASC II Express Professional IIe/IIc (Prodos or DOS3.3)..**$77**, Softronic's Softerm II.**$169**
Mousewrite IIe/c **$99**: Home Accountant Prodos IIe/c..**$39**: Flight Sim. IIe/c...**$35**
New!! Managing Your Money by Andrew Tobias (for 128K Apple **IIe/IIc**).....**$119**
New **FANTA VISION IIe,c**(Create Your own Graphics with actions,Cartoons, etc).**$39**
PRINTSHOP...**$35**: GRAPHICS DISK I/II.**$19 Ea.**, **Color PAPER(120)+Envelope(42)...$19**
PRINTSHOP COMPANION..CALL: DAZZLEDRAW 128K IIe/IIc.**$45**: SIDEWAYS.....**$35**
SCIENCE TOOLKIT IIe,IIc, II+(Hardware & Software Master Module by Broderbund).....**$49**
NEWSROOM.**$39**: CLIPART for NEWSROOM**$29**: **MATHBLASTER $35**: **ALGEBLASTER**...**$35**

WE ALSO CARRY EDUCATIONAL SOFTWARE FROM WELL KNOWN PUBLISHERS SUCH AS ARTSCI, COMPRESS, EARTHWARE, FOCUS MEDIA, GESSLER, INTELLECTUAL, J & S, K-12 MICROMEDIA, MECC, ROGER WAGNER, SCHOLASTIC, & SOUTHWESTERN. SCHOOL PURCHASE ORDERS/ENQUIRIES WELCOME.

WRITE US FOR A DETAILED PRICE LIST. AZ RESIDENTS, ADD SALES TAX. $3 shipping/$100. SPECIFY MACHINE TYPE => IIe, IIc, II+, or MAC. ALLOW 2 WEEKS for PERSONAL CHECKS. VISA/MC/AMEX OK.
Note: Price(s) subject to availability and expire on 3/31/86. **Adam: II Ad : 12-85 Nibble Ad:1-86**

COMPUTERS PLUS COMPANY

2303 N. 44TH ST. #11 PHOENIX AZ 85008

PHONE (602) 955-1404

MasterCard VISA AMERICAN EXPRESS

CIRCLE NUMBER 68

LISTING 2: LIGHT.PEN.DEMO

```
1   REM ************************
2   REM * LIGHT PEN DEMO       *
3   REM * BY DAVID GAUGER II   *
4   REM * COPYRIGHT (C) 1986   *
5   REM * BY MICROSPARC, INC   *
6   REM * CONCORD, MA  01742   *
7   REM ************************
100   REM  *** INITIALIZE ***
110   GOSUB 190
120   PRINT  CHR$ (4);"BLOAD PEN.DRIVER"
130 PEN = 770: REM    LOCATION OF  MACHINE LA
    NGUAGE DRIVER
140 V = 768:H = 769: REM         VERTICAL (CO
    LUMN) AND          HORIZONTAL (ROW) COORDI
    NATE  LOCATIONS
150   GOTO 210
160   REM  *** SUBROUTINES ***
170   INVERSE : PRINT " ";: NORMAL : RETURN : REM
      SUBROUTINE  TO PRINT AN INVERSE SPACE
      (BLOCK) ON THE SCREEN
180   PRINT " ";: RETURN : REM      SUBROUTINE
    TO PRINT A SPACE
190   TEXT : HOME : RETURN
200   REM  *** MAIN PROGRAM ***
210   HOME : HTAB 5: PRINT "NIBBLE LIGHT PEN D
    EMONSTRATION": PRINT "** COPYRIGHT 1986
    BY MICROSPARC, INC.**"
220   VTAB 8
230   PRINT "THIS PROGRAM IS MEANT TO DEMONSTR
    ATE"
240   PRINT "ONE POSSIBLE USE FOR THE NIBBLE L
    IGHT"
250   PRINT "PEN IN YOUR OWN PROGRAMS."
260   PRINT : PRINT
270   PRINT
280   PRINT "PLEASE NOTE THAT THE PROGRAM IS N
    OW"
290   PRINT "WAITING FOR YOU TO TOUCH THE INVE
    RSE"
300   PRINT "SPACE WITH THE LIGHT PEN."
310   VTAB 23: HTAB 5
320   PRINT "TOUCH PEN HERE TO CONTINUE ==>";
330   GOSUB 170: REM  PRINT         INVERSE SPA
    CE
340   CALL PEN: REM    PROGRAM       CONTROL PA
    SSED TO LIGHT      PEN DRIVER WHICH RETUR
    NS    ONLY WHEN IT FINDS PEN
350   REM  *** MENU SCREEN ***
360   GOSUB 190: HTAB 5: PRINT "NIBBLE LIGHT P
    EN DEMONSTRATION"
370   VTAB 8: HTAB 12: GOSUB 170: GOSUB 180
380   PRINT "CATALOG DISK"
390   VTAB 11: HTAB 12: GOSUB 170: GOSUB 180
400   PRINT "RING BELL"
410   VTAB 14: HTAB 12: GOSUB 170: GOSUB 180
420   PRINT "LIST PROGRAM"
430   VTAB 17: HTAB 12: GOSUB 170: GOSUB 180
440   PRINT "END"
450   CALL PEN
460   REM  * LIGHT PEN DECODING *
470   IF  PEEK (V) = 7 THEN  PRINT  CHR$ (4);"
    CATALOG": FOR X = 1 TO 2000: NEXT X: GOTO
    360
480   REM  NOTE THAT THE DRIVER   RETURNS (V)
    ONE LESS THAN     THE VTAB VALUE OF THE
    SAME     SPOT
490   REM   NOTE ALSO THAT WE DONT HAVE TO DEC
    ODE THE HORIZ.  VALUE IN THIS CASE BECAU
    SE   THE VERTICAL VALUE ALONE   IS ENOUG
    H
500   IF  PEEK (V) = 10 THEN  PRINT  CHR$ (7) +
     CHR$ (7) +  CHR$ (7) +  CHR$ (7) +  CHR$
    (7);: GOTO 360: REM  5 BELLS
510   IF  PEEK (V) = 13 THEN  HOME : LIST : HOME
    : GOTO 360
520   REM    IF (V)=16 THEN THE    PROGRAM WIL
    L FALL THROUGH    TO THIS POINT
530   END
```

END OF LISTING 2

Who says good software has to be expensive?

Now you can buy a full function spreadsheet for $49.95 or less. IACcalc.™

Written for your Apple,® IACcalc works with most 80-column boards and supports RAM cards in multiple combinations up to 512K. Reads VisiCalc® models and stores data on hard disk. It also allows individual column widths for text or data entries.

IACcalc has easy-to-use menu screens. No more blank stares from your monitor leaving you guessing which way to go. Just press the ESC key and pick the option number you wish to use. That's it.

But the most important feature of all won't be found on any of the menus. Our price. Only $49.95 retail. Or $39.95 if you're a member of the International Apple Core.

Now, the best way to find out if IACcalc is all we say it is—is to try it. That's why we offer a 30 day money-back guarantee.

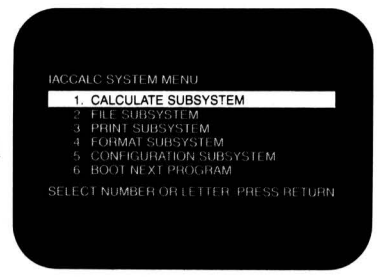
Order your copy today and when you receive it, try it on your spreadsheet problems for 30 days. Then if you decide not to keep it, return it and get all your money back.

This is just one example of the many benefits available through the International Apple Core, an organization that has served Apple users and user groups since 1979. We would be glad to send you full details about us—just ask.

To order your copy of IACcalc, use your VISA or MasterCard and call (408) 727-7713.

By mail, send check or money order to: International Apple Core, Product Dept #N-03, 908 George Street, Santa Clara, CA 95054. Payment made to International Apple Core must accompany order and must be in U.S. funds drawn on U.S. bank. Include $2.50 for shipping. California residents add 6.5% sales tax. Airmail outside U.S., Canada and Mexico add $11.00 shipping.

Please allow 2 to 3 weeks for delivery.



IACcalc System Menu defines subsystems.

INTERNATIONAL APPLE CORE™

CIRCLE NUMBER 52

Q.A. ASK NIBBLE!

by Cecil Fretwell

Q. Under ProDOS, I like for my STARTUP file to show the volume name. Therefore, my STARTUP programs usually include a line like 10 PRINT "/MY.DISK/". Recently, I typed 10 /MY.DISK/ instead. When I booted the system, it "went South" (it died). What's wrong with ProDOS?

A. All software developers try to write perfect programs, but they suffer occasional lapses. Such is the case with the ProDOS developers at Apple Computer. In a running Applesoft program, the BASIC.SYSTEM file loaded in a normal boot situation has a problem. It always traps the first byte in an Applesoft program line by making the Applesoft ROM firmware think TRACE mode is in effect. Unfortunately, the system code assumes the first byte in a line is a variable or a token.

In your case, the system went into an infinite loop trying to process the first byte in the line (the slash character). If you are really curious, try various combinations on a disk you can afford to lose. For instance, 10 OR as the first line will make the system go dead with the "in use" light active on the drive.

Fixing this bug would require a change in BASIC.SYSTEM that would only be valid for one particular version. Apple will probably fix it in a later release. But in the meantime, just be very careful when you key in an Applesoft program.

Q. Recently, I switched from DOS 3.3 to ProDOS. Having some knowledge of assembly language programming, my normal assembly code for PRINT CHR$(4);"*string*" sends $84 through COUT ($FDED) followed by the string. It doesn't work under ProDOS.

A. Obviously, ProDOS handles this situation differently from DOS 3.3. Under ProDOS, don't send the CHR$(4) processing through COUT. Instead, place the string in the $200 keyboard buffer terminated by a carriage return ($D). Then perform JSR DOSCMD ($BE03). If the carry flag is returned clear, your "command" was executed successfully. If the carry flag is set, you have an error and the Accumulator contains the error code, such as $10 for a SYNTAX ERROR.

This does not solve all your problems, though. Apple Computer sends technical notes to authorized developers when they release a new system. Their Technical Note 2 shows that CATALOG, CAT, PREFIX, CREATE, RENAME, DELETE, LOCK, UNLOCK, VERIFY, SAVE, STORE, RESTORE, PR#, IN#, FRE, BYE, OPEN, CLOSE, FLUSH, POSITION, BRUN, BLOAD and BSAVE work correctly from machine language and return control to the calling routine. On the other hand, the hyphen (-), RUN, LOAD, CHAIN, READ, WRITE, APPEND and EXEC do not work correctly and/or do not return control to the calling routine. In most of these cases, there is no easy fix without modifying BASIC.SYSTEM.

Q. Under assembly language control, I want to LOAD a program, then RUN it. Placing RUN followed by a carriage return ($8D) in the keyboard buffer at $200, then using DOSCMD ($BE03) produces some strange effects. Is there a solution?

A. As explained in the answer to the previous question, RUN is one of the commands that doesn't work correctly. However, after limited testing, I would say there is a solution. You are on the right track but need some additional code. Issuing a RUN through DOSCMD places you in the running state. However, at the end of DOSCMD the carry flag is set and the Accumulator contains the Applesoft SYNTAX ERROR code ($10).

Don't worry about the error code. After you use DOSCMD to execute the RUN, perform a JSR $D664. This initializes an Applesoft program with regard to variable pointers, and then performs a JMP $D7D2 to start a new program.

Q. Under DOS 3.3, when an Applesoft program requires a large chunk of memory for an assembly language subroutine, I can lower HIMEM by using the HIMEM statement in the program or storing the desired value in $73 and $74. I recently CONVERTed to ProDOS a program that had its HIMEM at location 38208 ($9540). When I executed the program, it bombed. Why does the program work under DOS 3.3 but not ProDOS?

A. You are violating a very important ProDOS rule. The system always assumes that HIMEM points to a page boundary, i.e., that $73 is always zero. Change HIMEM :38208 to HIMEM:38144 ($9500) to set things right.

By the way, I don't like to move HIMEM in this fashion because it assumes that the string storage area in memory is empty. If I want to tuck some code between string storage and the system code, I first load the code into a safe location in memory such as $2000 and execute it. I load the Accumulator with the number of memory pages required, then issue a JSR GETBUFR ($BEF5). (Apple documents this procedure in technical notes it releases to authorized developers). If the carry bit returns set, the system could not allocate the amount of memory requested. If the carry bit returns clear, string storage has been safely shifted down in memory and the Accumulator contains the beginning page of the allocated memory. I use this value to move the desired code from the $2000 area to the allocated memory.

Apple expert Cecil Fretwell answers your tough questions on topics ranging from ProDOS and printing to the internals of the Apple. Send your questions to: Ask Nibble, 45 Winthrop St., Concord, MA 01742. We can't promise to answer all of them, but we're sure that you'll find a wealth of new information in the ones we publish!

The Apple According to Mossberg

Sandy Mossberg has been exploring, dissecting and demonstrating the machine language that lies beneath Applesoft and DOS for over three years. His Disassembly Lines column has educated and entertained thousands of fascinated Nibble readers. And now the first two years of his work are available in special book and disk packages.

Disassembly Lines Volume 1 is a collection of the first eight installments of Sandy's column. You'll learn how to use Applesoft machine language in your own programs; how to build an Applesoft Line Editor; how Applesoft handles hex/dec data conversion, program listing and keyboard buffers; and how to make full use of the extended memory of your 64K Apple.

D/L Volume 1 includes a disk full of demonstration programs that show the routines in action, plus an unpublished bonus: source files (in Big Mac/Merlin format) for each of the major tables in Applesoft and DOS 3.3!

Disassembly Lines Volume 2 presents the next eleven installments of Disassembly Lines. Sandy delves deeply into DOS 3.3, divulging base conversion routines, keyboard and video intercepts, and command parsing and processing. A complete discussion of the DOS 3.3 File Manager examines each of the DOS commands in fine detail.

D/L Volume 2 comes complete with a disk containing all disassemblies and demonstration programs.

Order today from Nibble!

Send me Disassembly Lines!

☐ Volume 1 – $29.95 Postpaid
☐ Volume 2 – $29.95 Postpaid
☐ Save! Both volumes for only $50.00 Postpaid

Payable in U.S. funds only.

☐ MasterCard
☐ Visa
☐ Check/M.O.
(Mass. residents add 5% sales tax)

Name _____ Tel. # _____

Address _____

City _____ State _____ Zip _____

Signature _____ Date _____

Charge Card # _____ Exp. Date _____

nibble 45 Winthrop St., Concord, MA 01742 (617) 371-1660

FIRST TOTALLY PORTABLE //c*

Take it with you everywhere!

System Features:

- 24 Line by 80 Character LCD Display
- 8 Hour rechargeable power supply
- 20 lb. total system weight including your //c
- Durable cordura carrying case with handle and shoulder strap

C-VUE Flat Panel Display $475.00
PRAIRIE POWER battery & case 190.00

TOTAL if purchased separately $665.00
if purchased together SAVE 70.00

PORTABLE SYSTEM PRICE
TOTAL $595.00

For more details and ordering information call 1-800-GET-CVUE (outside California) or (619) 274-1253 (inside California). MasterCard and VISA accepted. Prices subject to change without notice.

ROGER COATS
P.O. Box 171466
San Diego, CA 92117

*Apple //c not included
Apple //c is a registered trade mark of Apple Computer Inc.

CIRCLE NUMBER 70

ON THE SCENE

Edited by Mary Locke

LOCKSMITH 6.0 is a new version of the powerful bit-copy program that also recovers crashed disks, restores deleted files and performs hardware diagnostics. Compatible with Apple // series computers, it includes a framing-bit analyzer, an automatic boot tracer, a sector editor and a disk with procedures to copy hundreds of Apple programs. The price is $79.95. Contact: **Alpha Logic Business Systems, 4119 North Union Rd., Woodstock, IL 60098, (815) 568-5166.**

CIRCLE NUMBER 120

GUITAR MASTER is a self-paced tutorial program that demonstrates almost 400 chords, 375 progressions, 18 picking and strumming patterns and all the major scales for the guitar. It includes a program for tuning and a guide to transposing. GUITAR MASTER is designed for ages 10 to adult and requires a Mockingboard and an Apple II, II Plus or //e. Its price is $49.95. Contact: **MasterSoft, P.O. Box 1027, Bend, OR 97709, (503) 388-7654.**

CIRCLE NUMBER 121

VOTALKER AP's board and software translators enable your computer to speak in an unlimited vocabulary with 32 inflections, to sing in five octaves with over 4,000 pitch settings and to generate sound effects. It has two preprogrammed voice modes that can be varied by an on-board filter. It works with the Apple II, II Plus and //e and costs $179. Contact: **Votrax, Inc., 1394 Rankin Rd., Troy, MI 48083, (313) 588-2050.**

CIRCLE NUMBER 122

BEACH-HEAD and BEACH-HEAD II are strategy/action games for the Apple // series. They feature graphics, animation, sound effects, and multiple scrolling play screens. High scores can be saved and to help you get them, demonstration modes let you practice on various phases of the games. Both programs require a joystick and are $34.95 each. Contact: **Access Software, 2561 South 1560 West, Woods Cross, UT 84087, (801) 298-9077.**

CIRCLE NUMBER 123

PROFILER 2.1 is an easy-to-use advanced file management system that lets you file, sort, calculate and print data, and allows you to design custom formats quickly. Eight times faster than the DOS 3.3 version, this ProDOS version stores up to 1,500 records on 5 1/4 inch disks and up to 65,000 records on hard disk systems. It runs on any Apple // series computer with 64K and is priced at $89.95. Contact: **PM Software, 19731 Providence Lane, Huntington Beach, CA 92646, (714) 963-2221.**

CIRCLE NUMBER 124

GARRY KITCHEN'S GAMEMAKER: THE COMPUTER GAME DESIGN KIT lets you create games using five powerful tools: SpriteMaker, SceneMaker, SoundMaker, MusicMaker and the Editor. A joystick is used to select menu commands to create and animate characters, draw backgrounds, create sound effects and compose musical scores to incorporate in the games. Games can be saved and played independently of the master program. The program runs on Apple // series computers and is priced at $49.95. Contact: **Activision, Inc., 2350 Bayshore Frontage Rd., Mountain View, CA 94043, (415) 960-0410.**

CIRCLE NUMBER 125

SERIAL GRAPPLER+ is a printer interface card featuring built-in screen print commands to allow Hi-Res or double Hi-Res graphics or text to be printed with a few keystrokes. It allows you to print graphics in inverse, rotated, enlarged or in emphasized mode. SERIAL GRAPPLER+ is compatible with Imagewriters I and II, and the Scribe. The price is $119. An optional buffer, Bufferpack, is available with 16K, 32K or 64K. Contact: **Orange Micro, 1400 N. Lakeview Ave., Anaheim, CA 92807, (714) 779-2772.**

CIRCLE NUMBER 126

TAXPAK-86 is a menu-driven program for preparation of the 1985 form 1040 and Schedules A, B, C, D, G, SE and W. As each entry is made, all affected lines are updated and information is transferred from form to form. It always displays your current tax liability as well as the impact of the most recently entered transaction, so you can explore alternative tax strategies. It runs on Apple // series computers with 64K. The price is $39.95. Contact: **AHWARE, 805 Luz Court, Danville, CA 94526, (415) 837-7346.**

CIRCLE NUMBER 127

P.A.C.K. (Programmer's Assembly-Language Construction Kit) explains how to create, use and update subroutine libraries. It provides relocatable subroutines plus Editor, Assembler and Linking Loader utilities that you can combine into your own programs. P.A.C.K. is written in machine language for the Apple // series. Its price is $49.95 plus $3.00 shipping; specify DOS 3.3 or ProDOS. Contact: **Interactive Arts, 2715 Porter St., Soquel, CA 95073, (408) 475-7047.**

CIRCLE NUMBER 128

FANTAVISION is a special effects and animation generator for Apple // series computers with 64K. It uses techniques called "tweening" — creating smooth motion by generating 64 intermediate positions between objects, and "transformation" —

creating sequences in which an object in one frame is transformed into something else in the next. Your cartoons or movies can be saved on a self-booting disk. FANTAVISION is $49.95. Contact: **Broderbund Software, 17 Paul Dr., San Raphael, CA 94903, (415) 479-1170.**

CIRCLE NUMBER 129

THE INVOICER does batch or real-time processing, optional or automatic sales tax calculations, line extensions and totaling, invoice numbering, audit controls, and daily sales reporting with subtotals. It tracks taxable and nontaxable sales, has user-defined smart keys and allows access of account and stock number databases while invoicing. It runs on a 64K Apple II Plus, //e or //c and is priced at $69.95. Contact: **MiccaSoft, 406 Windsor Lane, New Braunfels, TX 78130, (512) 629-4341.**

CIRCLE NUMBER 130

BATTLE OF ANTIETAM, NORWAY 1985, and U.S.A.A.F. are strategy games. BATTLE OF ANTIETAM recreates the 1862 Civil War battle. NORWAY 1985 is an advanced-level game about the Soviet occupation of Norway and NATO's counter-attack. U.S.A.A.F. (United States Army Air Force) is a simulation of daylight bombing of German industry from 1943-1945. The programs are all one- or two-player games and run on the Apple // series. Their prices are $49.95, $34.95 and $59.95, respectively. Contact: **Strategic Simulations, Inc., 883 Stierlin Rd., Bldg. A-200, Mountain View, CA 94043, (415) 964-1353.**

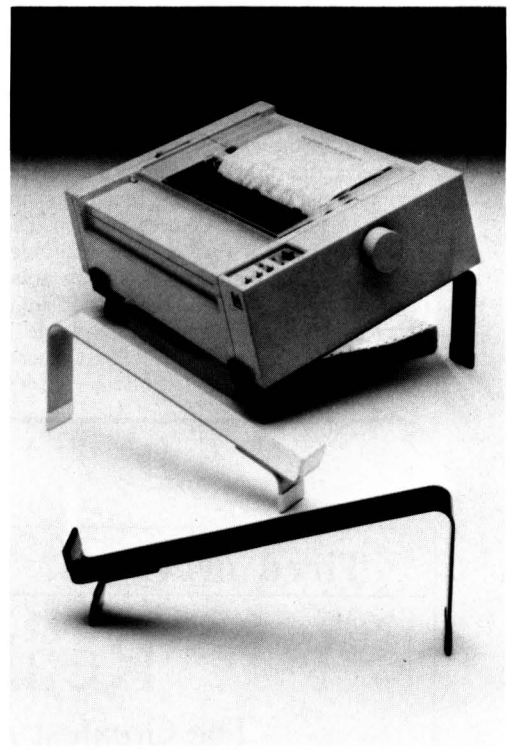
CIRCLE NUMBER 131

STICKYBEAR PRINTER is a graphic design and print program that makes it easy for children to create cards, stationery, posters, full-screen and half-screen pictures, and banners. Dozens of background and border patterns, hundreds of decorations and 10 fonts are included. It runs on the Apple // series and works with the Imagewriter II and Scribe, in addition to most dot matrix printers. It is priced at $39.95. Contact: **Optimum Resource, Inc., Station Place, Norfolk, CT 06058 (203) 542-5553.**

CIRCLE NUMBER 132

STAND UPS raise and tilt your printer for easy viewing of what's being printed and to provide space to store paper below the printer. Made of metal, STAND UPS have molded plastic feet that protect your printer and desktop. They are available in black and white. The price is $18.95 on direct orders. Contact: **Computer Cover Company, P.O. Box 3080, Laguna Hills, CA 92654 (714) 380-0085.**

CIRCLE NUMBER 133



PROFESSOR HALLEY'S COMET PURSUIT indicates the elevation and direction of Halley's comet at any time from any place on earth, making it easier for you to track it in the sky. Using both text and graphics, the program shows the correct azimuth and altitude. Background stars are plotted relative to the comet's position. The program includes a trivia game with over 500 questions on topics including solar science, comets, Edmond Halley and astronomy terminology. It runs on the Apple // series and is priced at $39.95. Contact: **Olympus Educational Software, 1660 Hotel Circle N., Suite 310, San Diego, CA 92108, (619) 296-8475.**

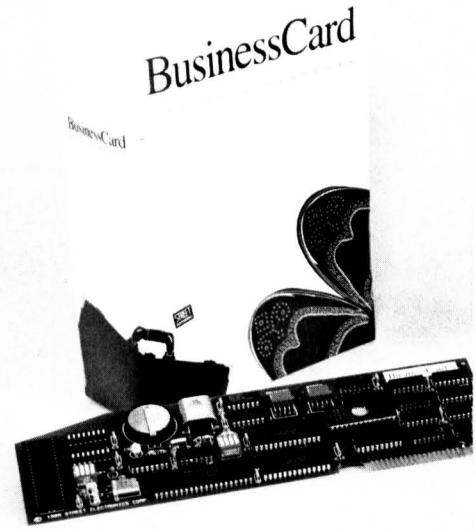
CIRCLE NUMBER 134

NOW THAT YOU KNOW ASSEMBLY LANGUAGE: WHAT YOU CAN DO WITH IT? is a book for people with some knowledge of 6502 assembly language. It explains how to develop a library of subroutines, how to access the 6502 stack, and how to minimize your coding by using the Apple's built-in routines. It is priced at $19.95. A disk containing the source code and one containing the object code for the book's programs are $15 each. Contact: **Relig Systems, Inc., 2068 79th St., Brooklyn, NY 11214, (718) 232-8429.**

CIRCLE NUMBER 135

BUSINESSCARD includes two serial interfaces and a clock/calendar with battery backup. It allows you to print text, and black-and-white or color graphics. It has over 60 built-in commands for clock, modem, and printing applications. It can drive a serial printer (including the Imagewriter II), and operate a 300 or 1,200 baud modem. It works with an Apple II Plus or //e. //e users are able to access commands from pull-down menus. The price is $219.95 (includes printer cable). Contact: **Street Electronics, 1140 Mark Ave., Carpinteria, CA 93013, (805) 684-4593.**

CIRCLE NUMBER 136



APPLEWORKS — THE PROGRAM FOR THE REST OF US answers the most commonly-asked questions about how to use the AppleWorks program. Written by Michael Sloan, who provides technical support for Apple Computer, Inc., the book guides you through all the major commands and options for the word processing, database and spreadsheet applications. It explains how to use AppleWorks with other programs and includes dozens of screen displays and examples. Contact: **Scott, Foresman and Company, 1900 E. Lake Ave., Glenview, IL 60025, (312) 729-3000.**

CIRCLE NUMBER 137

PROGRAMMERS: New DAVID-DOS II™

HIGH-SPEED DOS 3.3
DOS-MOVER/FAST-GARBAGE

Makes high-speed disk access under all conditions. DOS-mover frees 10,000 bytes of extra memory for programs. FRE command does ultra-fast garbage collection. Clock or manual disk dating. May be licensed.

100 Sectors In 7 Seconds

Speed Load/Save Applesoft, Binary & Integer 100 sector programs in 7 seconds. Tload/Tsave Random & Sequential Text Files at same speed. Speeds up programs like Home Accountant.

10K More Memory

Use HIDOS command in hello program for turnkey startup, adding 10K free memory to run 30% larger Applesoft programs than ProDOS. Moves DAVID-DOS II and 4 buffers above main memory.

Ultra-Fast Garbage

New FRE command collects a memory full of 6000 strings that are half garbage in two seconds. DOS 3.3 takes 12 minutes. FRE is so fast it is not noticeable during run of most programs.

Clock Dating

Automatic date stamping of disk files is set up for 6 kinds of clocks or Hello manual date entry. New DATE command will auto-insert date in correspondence.

All times in seconds. (Time Test programs available)		DAVID DOS-II	ProDOS	DOS 3.3
TEXTFILES (100 Sectors)	TSAVE	8.0	NO	NO
	TLOAD	6.2	NO	NO
(791 Strings,	WRITE	29.3	28.0	88.4
32 chars ea)	READ	24.3	16.3	83.8
	PRINT/READ	44.2	45.9	117.1
(442 Sectors, 7 x 500)	APPEND	142.3	142.9	1231.2
APPLESOFT (100 Sectors)	*SAVE	6.4	16.4	33.1
	LOAD	5.0	4.0	23.5
INTEGER (100 Sectors)	*SAVE	6.6	NO	33.4
	LOAD	4.9	NO	23.4
BINARY (100 Sectors)	*BSAVE	7.3	18.4	28.7
	BLOAD	5.8	4.8	24.5
48K PROGRAM SPACE	APPLESOFT	36,352	NO	36,352
(With 3 Bufs avail)	INTEGER	36,352	NO	36,352
	BINARY	36,352	34,816	36,352
64K PROGRAM SPACE	APPLESOFT	46,592	32,256	35,756
(With 4 Bufs avail)	INTEGER	46,592	NO	35,756
	BINARY	46,592	41,728	35,756

*Add 5 seconds for Verify. Apple II, Applesoft & ProDOS are trademarks of Apple

Ten New DOS Commands

1. HIDOS moves DOS above 48K memory.
2. FRE makes ultra-fast garbage collection.
3. DATE stamp files. Clock or manual dating.
4. TLOAD speed loads all Text Files to mem.
5. TSAVE speed saves Text Files from memory.
6. TLIST lists all Text Files to screen/printer.
7. DUMP Binary/Ascii to screen or printer.
8. DISA disassembles Binary to screen/printer.
9. AL prints program Address & Length.
10. / is a one keystroke Catalog.

Identical Operation

All new commands operate identical to old DOS commands on the keyboard & in programs.

Install In Three Seconds

Install DAVID-DOS II on your full disks in three seconds without touching the programs. Create bootable high-speed new disks with a Basic, Binary, or Exec Hello & 35/40 tracks. Create Data-Disks with 30 extra sectors.

Variable Speed Scrolling

Key operated variable speed scrolling for TLIST, DUMP and DISA. Lower-case accepted on all commands. Catalog shows Free-Space. Automatic support of Integer or Applesoft Card in any slot, while in HIDOS or LODOS. Vendor license includes protection system.

Compatible

All DOS entry addresses have been preserved. DOS is same length and compatible with most software. David-Dos is fully copyable. Init areas were used for David-Dos. Works with all Apple IIs including IIe, IIc, Franklin & Basis with 48K/64K/128K & Corvus & Xebec Hard Disks. Complete documentation and many utilities are on the disk.

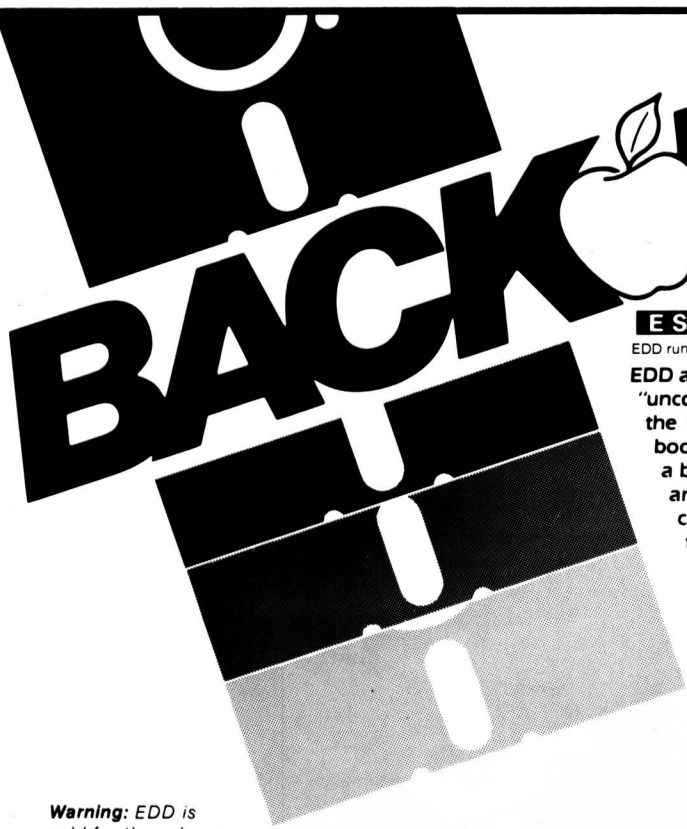
Add $2.00 Shipping. Overseas add U.S. $4.00. Calif. add 6%. **$39.95**

To Order: Send Check or Phone Visa/MasterCard.

DAVID DATA™

12021 WILSHIRE BLVD., SUITE 212B
LOS ANGELES, CA 90025 (213) 478-7865

CIRCLE NUMBER 72



NOW AVAILABLE AT YOUR LOCAL COMPUTER STORE

YOUR DISKS

ESSENTIAL DATA DUPLICATOR III™

EDD runs on Apple II, II plus, IIe, IIc and Apple III (in emulation mode) using one or two disk drives.

EDD allows you to easily and quickly make back up copies of your "uncopyable" Apple disks. ■ Since EDD has been preset to copy the widest range of copy-protections possible, you just simply boot up EDD, put the disk you want to copy in one disk drive and a blank disk in the other (EDD will work using one drive also) and in about 2 ½ minutes a copy is made. ■ Unlike the "copy-cards" which only copy "single load" programs, EDD copies the entire disk. This would be similar to hooking up two cassette recorders, playing from one, and recording to the other. ■ We have even included an option so you can check the speed of your disk drives because drive speeds running fast or slow can damage disks and cause other problems. ■ We publish EDD program lists (information about copy-protected disks) every couple of months, which EDD owners can receive. The current list is included with the purchase of EDD. ■ The bottom line is this; if EDD can't copy it, chances are nothing will.

$79.95 Ask for EDD at your local computer store, or, *to order direct:* send **$79.95** plus $2 shipping ($5 foreign). Mastercard/Visa accepted. *Prepayment required.*

***Warning:** EDD is sold for the sole purpose of making archival copies ONLY.*

UTILICO MICROWARE
3377 Solano Ave., Suite #352
Napa, CA 94558 (707) 257-2420

CIRCLE NUMBER 40

Products and Services Index

Circle #	Company Name	Page #
123	Access Software	140
125	Activision, Inc	140
75	Agranat Systems	146
127	AHWARE	140
*	ALF Products	147
49, 120	Alpha Logic Business Systems, Inc	110, 140
54	Analytical Solutions and Software	120
*	Applied Engineering	C2, 10-11, 19, 21
*	Applied Engineering	23, 91, 101, 113
50	Applied Technology Laboratories	111
55	Artificial Intelligence Research Group	120
48	Beagle Bros. Micro Software	105
58	Black Sun	125
56	Blankenship	121
129	Broderbund Software	140
61	Bullpenn Products	128
*	Business Computers of Peterborough	123
27	The Byte Works, Inc	67
77	Cardinal Point, Inc	147
43	Central Point Software, Inc	98
*	CH Products	1
78	Childsoft	147
31	Cirtech	107
20	Coit Valley Computers	53
59	Columbia Software	127
*	Compucrafts	119
6	Compuserve	24-25
79	Computer Accents, Inc	147
133	Computer Cover Company	141
44	Computer Network	99
68	Computers Plus Co	135
65	Computer Supply	133
*	Covox, Inc	46
82	Da Poma, Inc	147
72	David Data	143
*	Dickens Data Systems	98
64	Digital Images	133
83	Disk-of-the-Month Club	147
*	Diversified Software Research	33
66	Eclipse Systems	133
46	Extend-80	103
13	First Class Peripherals	37
38	Gilles Asselin	86
138	Groiler Electronic Publishing, Inc	145
12	Gutenberg Software, Ltd	36
47	Hardcore COMPUTIST	103
*	H & E Computronics	C3
74	High Order Microelectronics	145
60, 128	Interactive Arts	128, 140
28	Interactive Microware, Inc	68
52	International Apple Core	114, 137, 141
80	Isle Soft	147
9	Jameco Electronics	27
57	KWH Technology	121
69	Kyan Software	136
51	Laing Electronics	112
15	Legend Industries, Ltd	44
121	MasterSoft	140
32	Meca	72, 73
139	Mentor Learning Systems, Inc	145
130	MiccaSoft	141
7, 8	Micro Data Products	26
16	MicroDimensions, Inc	45
*	MicroSPARC, Inc	58, 39, 47, 139
19, 25, 42	MicroSPARC, Inc	50, 63, 95
67	Micro Systems Research	134
*	NIBBLE	
	Apple Secrets	66
	Express Volumes	80-81
	Express IV	96
	Express V	55
	Express VI	28
	MLE	90
	Subscription	64
	Software Catalog	156-159
17	Nibble Notch	45
22	Nikrom Technical Products, Inc	60
11	Northeastern Software	34, 35
29	Norwich Data Services	68
134	Olympus Educational Software	142
63	Optimal Technology	129
132	Optimum Resource, Inc	141
26, 126	Orange Micro	67, 69, 140
76	Order House 88	146
30	Panamax	69
18	Pinpoint Publishing	48-49
4, 124	PM Software	8, 140
21	Pro/Pac, Inc	54
10	Programs Plus	31
33	Quinsept, Inc	100
37	Quorum International, Unltd	85
3	RC Systems, Inc	7
71	Relational Systems Corp	142
135	Relig Systems, Inc	142
70	Roger Coats	139
24	Sav-Soft Products	119
137	Scott Foresman and Company	142
41	Sensible Software	92
73	Signal Computer Consultants	145
53	Silicon Express	117
5	Sir-Tech Software, Inc	9
62	Softinson Data Corporation	129
23	Software Discounters of America	61
*	Software Masters	62
*	Southern California Research Group	108
*	So What Software	136
36	Spies Laboratories	84
45	Star Micronics	102
81	STL	147
131	Strategic Simulations, Inc	141
84, 136	Street Electronics Co	C4, 142
2	Sweet Micro Systems, Inc	6
34	TermExec	100
*	Triad Software Products	70
83	Unlimited Software	147
40	Utilico Microware	143
35	Vilberg Bros. Computing, Inc	83
122	Votrax, Inc	140
14	Woodchuck Industries	44
1	X-10 (U.S.A.), Inc	4
39	Zephyr Services	87

Reader Feedback

Help us keep bringing you better Nibbles! Use the convenient Reader Service Card to rate the articles in this issue. The articles are listed below along with their article numbers.

Article	Article #
Turtle BASIC: J.B. Ward	1
Nibble Calculator: Newton Saiyuen Lee	2
Tank Combat: Rudy A. Guy	3
ProCursor: Jim Lazar	4
ProDOS Directory List: Keith Stattenfield	5
DISPLAY: Ken Manly	6
Games!: John DiPrete	7
Nibbling at Assembly Language Part V: S. Scott Zimmerman	8
ProDOS Output Processing: Sandy Mossberg	9
Nibble Light Pen: David Gauger II	10

Order Card — Use this convenient postpaid card for ordering Nibble subscriptions, books and software

☐ Customer Reference No. ***(Renewals Only)*** _____

Nibble Subscriptions:

☐ 12 issues (U.S.)	$26.95	_____
☐ 24 issues (U.S.)	$49.95	_____
☐ 36 issues (U.S.)	$69.95	_____
☐ U.S. First Class	$51.95	_____
☐ Canada Surface	$34.95	_____
☐ Canada Air Mail	$59.95	_____
☐ Outside U.S. & Canada Surface	$39.95	_____
☐ Outside U.S. & Canada Air Mail	$89.95	_____

I use the following computer(s): ☐ Apple // Family ☐ Macintosh

Nibble Express

☐ Volume I	$14.95	_____
☐ Volume II	$14.95	_____
☐ Volume III	$17.95	_____
☐ Volume IV	$17.95	_____
☐ Volume V	$19.95	_____

(Mass. residents add 5% sales tax.)

Add: $1.75/copy Shipping/Handling U.S.
$2.75/copy Surface Outside U.S.
$6.50/copy Air Mail Outside U.S.

Nibble Software
(See Software Catalog for Prices)

Disk	Price
☐ _____	_____
☐ _____	_____
☐ _____	_____
☐ _____	_____
☐ _____	_____

U.S. shipments add $1.50 for shipping and handling. Outside U.S. add $2.50 for the first disk ordered and $1.00 for each additional disk. (Mass. residents add 5% sales tax.)

Total cost of items ordered _____
Shipping/handling _____
Mass residents add 5% sales tax on software/book orders _____
TOTAL ORDER (Payable in U.S. funds only) _____

Please charge to my:

☐ MasterCard ☐ VISA (Check One)

Card No. _____ Exp. Date _____
Signature _____
Telephone _____

☐ Payment enclosed (check, M.O.)
☐ Bill me — U.S. subscription only
Payable in U.S. funds only

Name _____
Address _____
City _____ State _____ Zip _____
Country (if outside U.S.) _____

Free Product Information

THIS CARD VALID UNTIL MARCH 31, 1986

Reader Service: To receive information about the products advertised in this issue simply find the Reader Service number at the bottom of the ad and circle the corresponding number listed here. Please allow 4-6 weeks for the advertiser to send you the information requested.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100
101	102	103	104	105	106	107	108	109	110
111	112	113	114	115	116	117	118	119	120
121	122	123	124	125	126	127	128	129	130
131	132	133	134	135	136	137	138	139	140
141	142	143	144	145	146	147	148	149	150
151	152	153	154	155	156	157	158	159	160
161	162	163	164	165	166	167	168	169	170
171	172	173	174	175	176	177	178	179	180
181	182	183	184	185	186	187	188	189	190
191	192	193	194	195	196	197	198	199	200
201	202	203	204	205	206	207	208	209	210
211	212	213	214	215	216	217	218	219	220
221	222	223	224	225	226	227	228	229	230
231	232	233	234	235	236	237	238	239	240
241	242	243	244	245	246	247	248	249	250
251	252	253	254	255	256	257	258	259	260
261	262	263	264	265	266	267	268	269	270
271	272	273	274	275	276	277	278	279	280
281	282	283	284	285	286	287	288	289	290
291	292	293	294	295	296	297	298	299	300

Article Rating System

Here's your chance to let us know what you think of this issue of Nibble. To help us provide you with the best articles and programs, please rate each article by first looking at the list in the Products and Services Index and finding the article number. Carefully consider whether the article is useful to you and whether you would like to see more articles on similar topics. Then rate each article by circling the numbers below to indicate whether you think it is **Excellent, Good, Fair** or **Poor**. Your feedback can help us keep bringing you better Nibbles!

Article No.	1	2	3	4	5	6	7	8
Excellent	1	5	9	13	17	21	25	29
Good	2	6	10	14	18	22	26	30
Fair	3	7	11	15	19	23	27	31
Poor	4	8	12	16	20	24	28	32

Article No.	9	10	11	12	13	14	15	16
Excellent	33	37	41	45	49	53	57	61
Good	34	38	42	46	50	54	58	62
Fair	35	39	43	47	51	55	59	63
Poor	36	40	44	48	52	56	60	64

Article No.	17	18	19	20	21	22	23	24
Excellent	65	69	73	77	81	85	89	93
Good	66	70	74	78	82	86	90	94
Fair	67	71	75	79	83	87	91	95
Poor	68	72	76	80	84	88	92	96

☐ Circle #300 for a 12-issue subscription. We'll bill you for $26.95 (U.S. only).

Name _____
Address _____
City _____ State _____ Zip _____
(Country, if outside U.S.) _____

Free Product Information

THIS CARD VALID UNTIL MARCH 31, 1986

Reader Service: To receive information about the products advertised in this issue simply find the Reader Service number at the bottom of the ad and circle the corresponding number listed here. Please allow 4-6 weeks for the advertiser to send you the information requested.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100
101	102	103	104	105	106	107	108	109	110
111	112	113	114	115	116	117	118	119	120
121	122	123	124	125	126	127	128	129	130
131	132	133	134	135	136	137	138	139	140
141	142	143	144	145	146	147	148	149	150
151	152	153	154	155	156	157	158	159	160
161	162	163	164	165	166	167	168	169	170
171	172	173	174	175	176	177	178	179	180
181	182	183	184	185	186	187	188	189	190
191	192	193	194	195	196	197	198	199	200
201	202	203	204	205	206	207	208	209	210
211	212	213	214	215	216	217	218	219	220
221	222	223	224	225	226	227	228	229	230
231	232	233	234	235	236	237	238	239	240
241	242	243	244	245	246	247	248	249	250
251	252	253	254	255	256	257	258	259	260
261	262	263	264	265	266	267	268	269	270
271	272	273	274	275	276	277	278	279	280
281	282	283	284	285	286	287	288	289	290
291	292	293	294	295	296	297	298	299	300

Article Rating System

Here's your chance to let us know what you think of this issue of Nibble. To help us provide you with the best articles and programs, please rate each article by first looking at the list in the Products and Services Index and finding the article number. Carefully consider whether the article is useful to you and whether you would like to see more articles on similar topics. Then rate each article by circling the numbers below to indicate whether you think it is **Excellent, Good, Fair** or **Poor**. Your feedback can help us keep bringing you better Nibbles!

Article No.	1	2	3	4	5	6	7	8
Excellent	1	5	9	13	17	21	25	29
Good	2	6	10	14	18	22	26	30
Fair	3	7	11	15	19	23	27	31
Poor	4	8	12	16	20	24	28	32

Article No.	9	10	11	12	13	14	15	16
Excellent	33	37	41	45	49	53	57	61
Good	34	38	42	46	50	54	58	62
Fair	35	39	43	47	51	55	59	63
Poor	36	40	44	48	52	56	60	64

Article No.	17	18	19	20	21	22	23	24
Excellent	65	69	73	77	81	85	89	93
Good	66	70	74	78	82	86	90	94
Fair	67	71	75	79	83	87	91	95
Poor	68	72	76	80	84	88	92	96

☐ Circle #300 for a 12-issue subscription. We'll bill you for $26.95 (U.S. only).

Name _____
Address _____
City _____ State _____ Zip _____
(Country, if outside U.S.) _____

NO POSTAGE
NECESSARY IF
MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 43 CONCORD, MA

POSTAGE WILL BE PAID BY ADDRESSEE

Nibble Publications
45 Winthrop Street
Concord, MA 01742-9990

BUSINESS REPLY CARD
FIRST CLASS PERMIT NO. 72 DALTON MA

POSTAGE WILL BE PAID BY ADDRESSEE

nibble®
P.O. BOX 345
DALTON, MA 01227-9938

NO POSTAGE
NECESSARY IF
MAILED IN THE
UNITED STATES

BUSINESS REPLY CARD
FIRST CLASS PERMIT NO. 72 DALTON MA

POSTAGE WILL BE PAID BY ADDRESSEE

nibble®
P.O. BOX 345
DALTON, MA 01227-9938

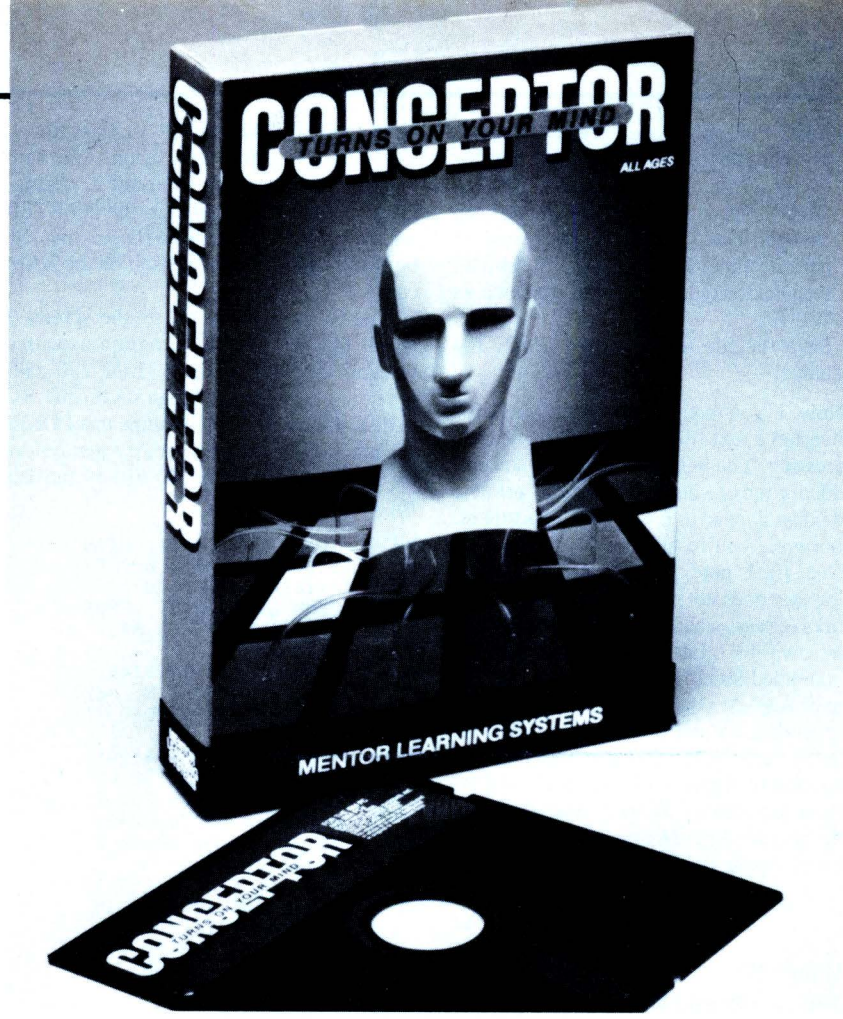
NO POSTAGE
NECESSARY IF
MAILED IN THE
UNITED STATES

THE INFORMATION CONNECTION combines an easy-to-use communications program, a text editor, and a tutorial featuring a simulated on-line session. It supports any modem that can be controlled with ASCII commands and allows you to send electronic mail or access networks. An on-screen clock displays elapsed time, and the program has an adjustable alarm and automatic shut-off to limit telecommunication costs. It works with 64K Apple // series computers and is priced at $59.95. Contact: **Grolier Electronic Publishing, Inc., 95 Madison Ave., New York, NY 10016, (212) 696-9750.**

CIRCLE NUMBER 138

CONCEPTOR is a challenging game-like program that helps players learn how to recognize, understand and classify information, and formulate concepts. From a set of 20 figures, players must choose 10 figures with similar characteristics. CONCEPTOR features colorful graphics, three levels of difficulty and 120 separate puzzles. It runs on Apple // series computers with 64K and is priced at $49.95. Contact: **Mentor Learning Systems, Inc., 1825 De La Cruz Blvd., Santa Clara, CA 95050, (408) 988-4114.**

CIRCLE NUMBER 139



continued from page 9

6. Enter the path name (prefix/filename) for each file to be converted. Do this one file at a time. Any file name that contained a space under DOS 3.3 has been changed; all spaces are converted to periods under ProDOS. The file should then load into memory as an Appleworks WP file.
7. Save the file to disk using a *new* file name.

Now, if you made no mistakes, you have converted a text file for AppleWorks word processing. You will have to change any embedded control characters for the printer, and you will need to reformat the document. Otherwise, you're all set.

One final note: Unfortunately, Apple Computer does not provide access to AppleWorks for early models of the Apple //. You must own an Apple //c or Apple //e with an extended 80-column card to take advantage of AppleWorks' features. However, an independent software developer, Norwich Data Services, Ltd., now provides a reconfiguration of AppleWorks for Apple II Plus owners (called Plus-Works), making it possible to run AppleWorks on a 64K, 80-column Apple II Plus.

Larry Solomon
Tucson, AZ

Higher FID

I enjoyed Donald Miller's article on moving FID to the RAM card ("Vigilant FID," Vol. 6/No. 6). However, I don't like running a loader program to install it. Changing the following lines in FID.CONVERTER will result in a version of FID.RC that can be BRUN and will install itself on the RAM card.

Line 30 prints a count on the screen (I didn't like the way the program appeared to hang as it did its stuff). Line 220 calls the new subroutine and then saves FID.RC to disk. Line 550 is a subroutine that POKEs the machine code into memory just before the image of FID. Lines 560-610 are the data for the machine code.

```
30 M = M + 1: VTAB 14: HTAB
   18: PRINT M:F = 0: IF
   M > 8983 THEN 140
220 GOSUB 550: PRINT  CHR$
   (4)"BSAVE FID.RC,A$17
   6A,L4854"
550 FOR I = 5994 TO 6146:
    READ N: POKE I,N: NEXT
   : RETURN : REM POKE I
   N RELOCATOR
560 DATA  141,129,192,141
   ,129,192,169,0,133,60
   ,133,66,168,169,255,1
   33,62,133,63,169,248,
   133,67,133,61,32,44,2
   54,160,18
570 DATA 185,210,23,153,1
   57,165,136,208,247,16
   9,70,141,241,168,169,
   73,141,242,168,169,19
   6,141,243,168,169,64,
   141,239,168,160
580 DATA 0,185,3,24,153,3
   ,208,200,208,247,238,
   169,23,238,172,23,173
   ,172,23,201,227,208,2
   34,32,47,251,32,88,25
   2,160
590 DATA 0,185,229,23,240
   ,6,153,0,4,200,208,24
   5,76,208,3,32,57,251,
   173,131,192,173,131,1
   92,32,3,208,173,129,1
   92
600 DATA 76,211,3,213,211
   ,197,160,167,198,201,
   196,167,160,198,210,2
   07,205,160,208,210,20
   7,205,208,212,160,212
   ,207,160,210,213
610 DATA  206,174,0
```

Kevin Sartorelli
New Zealand

Cleaner Heads

I have very much enjoyed your magazine ever since I started reading it about two years ago. Since then I have heard of people having trouble cleaning the disk drive heads on drive two. I have the solution. These POKEs turn on the drives (in slot 6), but they don't read or write on the disks:

POKE 49385,0	Turns drive on
POKE 49384,0	Turns drive off
POKE 49386,0	Selects D1
POKE 49387,0	Selects D2

James Faircloth
New Orleans, LA

nibble® software

To order Nibble programs on disk, use the convenient order card bound into this issue. See the last page of this catalog for ordering information and an additional ordering coupon.

Code:

NE __ : Featured in Nibble Express Volume __
V__ #__: Featured in Nibble Volume __ Issue # __

APPLE TRAC PLUS is an easy-to-use, adaptable system for analyzing "where your money goes." TRAC handles credit cards, checks, and cash and prints more than 10 different reports for budget spending analysis. Thousands are in use by *Nibble* readers. Companion programs TRAC BUDGET, TRAC GRAPHICS, and TRAC INCOME are offered as separate disks in this catalog. (NE 1)

SPACE MAZE puts you in control of a graphics space ship which you must guide through a maze under game paddle or joystick control. (NE 1)

Order Reference: [TR]. . .$24.95 + shipping

STAR ATTACK is a fast arcade space shooting gallery game. Friendly and hostile ships scan the skies. Zap the bad guys but don't hit your friends! It's not easy! (NE 1)

AIRSEA BATTLE pits the Army and Navy in low resolution graphics attack maneuvers. Choose sides and play a friend (or your Apple). (NE 1)

LOW RESOLUTION SHAPEWRITER helps you create and run high speed Lo-Res graphics shapes in your programs. Instructions show how to do the whole job. (V1 #2)

Order Reference: [SA]. . .$19.95 + shipping

APPLE TOUGH TEXT PROCESSOR is an affordable, line-oriented text processor. It's easy to use and features: upper/lower case, right margin justification, erase/retype, find/replace, shift paragraphs, parallel/serial printing, and a lot more. Learn how text processing works! (NE 1)

APPLE SIMON is a low resolution color graphics version of the popular Simon Says game. (NE 1)

TIGER GRAPHICS is a fast utility for printing graphics dumps to IDS 440/460 Paper Tiger printers. (NE 1)

Order Reference [TG]. . .$24.95 + shipping

F.A.S.T. is a forecasting and sales trend simulation and modeling program for testing different strategies in selling multiple products and services. Build a productivity model for testing critical hiring, pricing, etc. (NE 1)

SUPER WEAVER simulates an 8 harness loop in high resolution graphics. Simple keyboard commands provide control and the pattern can then be printed as a complete draw-down. (NE 1)

FOOTBALL simulates a hand-held game with alternate running, passing and kicking plays to score. (NE 1)

Order Reference [FW]. . .$24.95 + shipping

PERSONAL INVENTORY PROGRAM provides a complete disk based data management system for inventorying and tracking your personal assets. It is invaluable for insurance reporting. (NE 1)

AMPER INTERPRETER UTILITIES is a powerful utility library which adds a full function PRINT. . .USING command to Applesoft. It includes other commands for screen control. (NE 1)

FOUR IN A ROW pits you against another player to line up four low resolution graphics blocks in a row to beat your opponent. (NE 1)

Order Reference: [P2]. . .$29.95 + shipping

AUTOMATED INTELLIGENCE MAILING SYSTEM manages your mail lists with ease! You define up to 32 fields of information and control formatting, editing, sorting, and printing labels. You won't find another one like this. (NE 1)

CONCORDANCE is a powerful utility that cross references your Applesoft program variables and tells where they're used. (NE 1)

LOW SCORE II pits you against the Apple in Hi-Res graphics in a challenging dice game. (NE 1)

Order Reference: [AC]. . .$28.95 + shipping

WILL O' THE WISP takes you on an adventure through forests, caves and castles to the den of Ralph, the magician. (NE 1)

BLAST AWAY makes you a ghostbuster trying to zap the ghosts as the windows of the haunted house light up. (NE 1)

LOAN REDUCTION ANALYSIS projects the effect of interest changes and loan terms in analyzing major financing purchases. (V1 #8)

Order Reference: [WO]. . .$19.95 + shipping

CHAMP CHECKBOOK MANAGEMENT is a versatile check reconciliation system with editing, search, posting and running bank balances. (NE 2)

AMPER-READER is a utility that speeds the reading of sequential text files by a factor of 5-10 times. (NE 2)

APPLE COMMUNICATOR is a demonstration program for assembling words and sentences using the game paddle. It is adaptable for other inputs for use by the handicapped. (V2 #1)

Order Reference: [CH]. . .$28.95 + shipping

MUSIC RETRIEVAL SYSTEM is a flexible file management system oriented to cataloging and retrieving your record and tape collection. (NE 2)

LE MANS is a fast graphics racing game along twisting, narrowing roads. (NE 2)

MORSE CODE SYSTEM generates Morse code sight and sound for learning this communication method. (V2 #2)

Order Reference: [MR]. . .$19.95 + shipping

APPLE PAINT BOX is a Hi-Res graphics program that combines the functions of Etch a Sketch, Spiral Graph, and a function driven drawing board. Auto color fill-in selection and shape drawing make this an unusual drawing aid. (NE 2)

THE MAD, MAD, MAD CUBE is a low resolution graphics version of Rubik's cube. (NE 2)

APPLESOFT LINE FINDER locates individual program lines in memory and lets you append forbidden line numbers, repair programs, and more. (NE 2)

Order Reference: [PB]. . .$24.95 + shipping

STAR (STORAGE AND RETRIEVAL) SYSTEM is an extremely powerful data file management program for building your own custom databases and reports. (NE 2)

TRAP 'EM is an unusual text/graphics game that pits you against three deadly beasts which you attempt to trap before they eat you. (NE 2)

BOOT 3.2/3.3 is a "double boot system" for both versions of Apple's operating system. (V2 #4)

Order Reference: [ST]. . .$29.95 + shipping

DISK MASTER is a DOS 3.3 utility for recovering deleted files, catalog sorting, and direct track/sector examination. (NE 2)

TRAP/STEP single steps through the execution of an Applesoft program to examine program logic. It allows traps to be planted to examine specific variables during the program flow. (NE 2)

APPLE BIORHYTHMS produces graphics biocharts and printed reports of your biocycles. (NE 2)

Order Reference: [DM]. . .$29.95 + shipping

APPLE MARKETING AND PERSONAL SHOPPING helps keep track of your groceries and other items and even prints a pre-sorted shopping list. It lets you know what you have and what to get on your next visit to the store. (NE 2)

AMPER-JUMP and T-SORT is a fast machine language sorting utility for both primary and index fields in a file. (NE 2)

APPLE ARTIST includes functions for boxes, fill-ins, lines, and other Hi-Res graphics functions. (NE 2)

Order Reference: [MA]. . .$29.95 + shipping

APPLE DEBT/ASSET REPORTING SYSTEM is a home finance system for projecting future expenditures and debts. Project your net worth and test new investment and debt strategies. (NE 2)

AMPER-PRINT-USE simulates the PRINT... USING command for column heading, decimal and scientific notation, left and right justification and more. A unique Applesoft programming utility! (NE 2)

LASER BLASTER puts a host of aliens on a collision course with your space station. Rotate and fire before impact. (NE 2)

Order Reference: [DA]...$29.95 + shipping

FINANCIER III provides loan reduction calculation, amortization, two types of depreciation, and simple and compound interest. (V2 #7)

POUR BOY'S LE is an Applesoft program editing aid that includes start/stop list, fast cursor advance, shorten line length, and other functions. (NE 2)

APPLE CATSUP catalogs a disk and provides a shorthand menu method for issuing DOS 3.3 commands with one or two keystrokes. (NE 2)

NFL RATING SYSTEM measures relative strengths of all the NFL football teams and predicts the winners. (V2 #7)

Order Reference: [FI]...$19.95 + shipping

TRAC BUDGET is a flexible, fast budget preparation and reporting system for personal finances. It allows manual or automatic budget preparation and produces a variety of budget/actual report comparisons. It is recommended for use with TRAC PLUS. (NE 2)

TRAC GRAPH plots your finances in high resolution graphics, using information directly from the TRAC files. It requires TRAC PLUS. (NE 2)

APPLE ARCHIVES lets you back up your disk files automatically to tape cassette. (NE 2)

THREE-D TIC TAC TOE is a Lo-Res graphics version of the popular multi-level board game. (V2 #8)

Order Reference: [TB]...$29.95 + shipping

APPLE HEXPAD is a "phantom keypad" for fast, easy entry of machine language programs with automatic forward/backward spacing. (V2 #8)

DOS REMOVER physically removes DOS 3.3 from tracks one and two of your diskettes and frees up additional disk storage. (V2 #8)

COMMAND CHANGER lets you physically change and shorten your DOS 3.3 commands. (V2 #8)

SCREEN FORMATTER lets you customize data entry for professional looking screens. (V2 #8)

X-Y PLOT provides simple parameters for plotting data with Apple's DOS Tool Kit. (V2 #8)

Order Reference: [HX]...$19.95 + shipping

APPLE TURTLE GRAPHICS is a pseudo LOGO-type language for guiding your Hi-Res turtle around the screen! You write simple commands to move, turn, and draw and it follows your every command. (NE 3)

TRAC INCOME adds a third major module to the TRAC home finance system. It generates reports for year-to-date income profiles and monthly income statements, using data from TRAC BUDGET and TRAC files shown elsewhere in this catalog. (NE 3)

APPLESOFT LINE CRUNCHER automatically compresses multiple Applesoft program lines to smaller, multi-statement programs. A powerful program utility! (NE 3)

Order Reference: [TU]...$29.95 + shipping

APPLE TURTLEGRAPHICS TUTOR is a complete pseudo-LOGO turtle package for young and old! Dynamic demonstration library shows what the turtle can do and how to put it through its paces. Create your own pictures! Combine them! Even create 3-D effects. (NE 3)

Order Reference: [TT]...$23.95 + shipping

APPLE SCRAMBLE is a Hi-Res version of the popular letter/tiles game. Slide your way to success. (V3 #1)

APPLE CHECKER is an important utility for comparing your typing of *Nibble* programs with check sum totals published in *Nibble*. (V3 #1)

LO-RES HYPER-SCROLL provides fast 4-way scrolling of your screens. Produces spectacular screen displays. (V3 #1)

VARIABLE CROSS REFERENCER lets you extract, sort, and alphabetize your Applesoft variables together with program line numbers. (V3 #1)

AMPER FREE SPACE shows how much disk space is left...without loading other programs! (V3 #1)

Order Reference: [SC]...$19.95 + shipping

NIBBLE MACHINE LANGUAGE EDITOR is a unique utility for entering and editing machine language programs. It allows dynamic insert, delete, and exchange of machine code directly in memory. Includes a full disk command set for retrieving and saving files. A must for your library! (NE 3)

LAMP performs: decimal to hex conversion; parameter lists for machine language; locations of Applesoft lines; ASCII conversion; and free disk sectors. (V3 #2)

APPLE CAL lets you schedule events, holidays, and appointment days months or even years in advance. Prints calendars through the year 2000. (NE 3)

Order Reference: [ML]...$29.95 + shipping

APPLE SLUGGER is a Hi-Res action baseball game. Pitch, bat, and field in your own home! (NE 3)

APPLE PIE CHART is a set of routines for Apple's DOS Tool Kit for Hi-Res pie charting with color labels and highlighting (requires the Apple DOS Tool Kit). (V3 #2)

AMPER-FIND searches out variables, strings and commands at machine language speed. (NE 3)

SPEED SORT speeds your string sorting by a factor of 60. Handles one- or two-dimensional arrays in a flash. (V3 #2)

Order Reference: [SL]...$19.95 + shipping

APPLE BOND MANAGER is a comprehensive program for analyzing alternative bond investments. Comparative gross yields, capital gains yields, current and after tax yields. (NE 3)

DOS 3+2 is a fast switching utility between DOS 3.2 and 3.3. (NE 3)

APPLE DARTS is a spectacular Hi-Res graphics dart game with two levels of play. (NE 3)

Order Reference: [BN]...$29.95 + shipping

AMPERSPEED improves DOS 3.3 sequential text file reading by nearly three to one. (NE 3)

LAMP II works with LAMP (offered elsewhere in this catalog) to add more utility functions, including three-letter commands for cataloging and displaying free sectors on a disk. (V3 #3)

PROGRAM LIST PROTECTOR prevents your Applesoft programs from being listed. Keys the program access to your own code word. (V3 #3)

3-D MAP demonstrates three-dimensional landscapes using Apple's DOS Tool Kit. (V3 #3)

Order Reference: [AM]...$22.95 + shipping

MICRO-CALC generates your own, customized finance programs for home and business. You set up your own accounting spreadsheet, specify data, comments, and formulas for up to 20 columns and rows. Then Micro-Calc creates your own Applesoft program to do the job. (NE 3)

APPLESOFT VARIABLE CRUNCHER compresses long variable names to one or two letters, and shrinks your program down to size. (NE 3)

APPLE LIFE sets up Hi-Res colonies of life cells and shows their life...growth...decline...and creation of other colonies. (NE 3)

Order Reference: [MC]...$29.95 + shipping

ELECTRIC ABACUS is a Lo-Res graphics version of the timeless oriental calculator. (V3 #4)

APPLE GO puts DOS 3.3 commands under your one or two keystroke control. Uses a single letter for the command and a single designator for each file name. (NE 3)

QUICKSORT lets you sort arrays 50-170 times faster than Applesoft. (NE 3)

APPLE INITIA is a unique utility for setting and resetting Applesoft arrays at blinding machine language speed. (NE 3)

Order Reference: [AB]...$22.95 + shipping

APPLESOFT LINE EDITOR is a powerful utility that helps you write and edit Applesoft programs. Auto line numbering, insert, delete, and other commands allow you to directly edit program lines without having to retype or recopy entire lines. A must for your library! (NE 3)

SPRINT SPEED READING is an automatic text/number generator that helps increase your reading speed by up to 100%. It has options for preschoolers up through adult level of reading. (NE 3)

OTHELLO is a Lo-Res graphics version of the popular board game. You can play against another

nibble software

player or your Apple! (NE 3)

Order Reference: [AL]. . .$29.95 + shipping

DDT is a set of fast utilities for relocating and embedding programs of above DOS 3.3 buffers. (V3 #5)

ELECTRIC HOURGLASS is a Lo-Res graphics hourglass with sound and multicolored sand. (V3 #5)

DISK DUMPER provides a fast, simple way to dump selected or all DOS 3.3 disk files to your printer automatically. (NE 3)

INPUT SCREEN BUILDER provides samples for professional data entry screens using simple data statements. (V3 #5)

SUPER WRITER formats Applewriter I text in pages for forward and backward review. (V3 #5)

Order Reference: [DD]. . .$22.95 + shipping

APPLESOFT RECORD COMMAND SYSTEM is an advanced data filing, editing, printing, and communications program. Build your own records and fields. . .search records. . .merge files. . .and send data over communications lines. (NE 3)

DISK COMMANDER shortens DOS 3.3 commands to three characters and catalog names to 1-2 digits. (NE 3)

APPLE ART GALLERY is a comprehensive collection of 33 different art forms. Each art form has multivariations for hours of entertainment in Hi-Res graphics. (NE 3)

Order Reference: [AR]. . .$29.95 + shipping

BINARY CLOCK is an accurate machine language timepiece in Lo-Res graphics. (NE 3)

HI-RES DUMP produces giant double-sized prints of Hi-Res graphics to the Epson MX80 with Graftrax. (NE 3)

SHOPPING LIST performs filing, editing, and display of your personal shopping items. (V3 #6)

PRINT USR formats big and little numbers in your Applesoft programs with automatic decimal alignment and more. (V3 #6)

COMPARE APPLESOFT is a valuable utility for comparing different versions of your programs for changes, additions, and deletions. (NE 3)

Order Reference: [BC]. . .$22.95 + shipping

AMP-L-SOFT is a whole library of utilities under ampersand control. Gives you machine language control of tones, bits, tag sort, substring search, and hex conversion. Adds machine language speed to your Applesoft programs. (NE 3)

ELECTRONIC MESSAGE CENTER is a message handler for displaying sequences of messages for home and business. (NE 3)

APPLE BOWL FOOTBALL is a fast moving text game that uses actual football statistics to produce a realistic simulation of the game. (NE 3)

Order Reference: [AP]. . .$29.95 + shipping

THE SHAPE is a powerful free form shape table creation program. Create your shapes in Lo-Res graphics and translate them into Hi-Res shape tables automatically. (NE 3)

LIGHTSABER BATTLE pits Darth and Luke in a slugfest in Lo-Res graphics. (V3 #7)

PRICE DEMO uses a product pricing problem to illustrate disk file indexing. (V3 #7)

APPLE SCROLLER puts right, left, and downward text scrolling at your fingertips in machine language. (V3 #7)

LOWER CASE WRITER lets you type upper and lower case characters to your Hi-Res screen. (V3 #7)

Order Reference: [SH]. . .$23.95 + shipping

APPLE RECIPE BOX makes cook-booking simple! Retrieve, edit, and display, and print your favorite recipes with automatic portioning of different servings. (NE 3)

APPLE DISK DOCTOR is a comprehensive "zap" utility for direct reading, writing, and editing of your disks. See how DOS stores your data! (NE 3)

QUASAR II puts you in command of a starship charged with clearing distant galaxies, meteors, and asteroids. A *Nibble* version of the popular "Asteroids" game. (NE 3)

Order Reference: [RB]. . .$29.95 + shipping

FAT GRAPH gives you Hi-Res graphics tracking of your dieting objectives and weight lost. (V3 #8)

SOUTHERN DRAW is a new precision "pencil" cursor for Hi-Res graphics. (V3 #8)

SAV-DEL lets you save-delete selected Applesoft arrays without having to clear all of memory. (V3 #8)

FINANCIER/TAX DEPRECIATION adds new tax rules and updates for the economic recovery act. (V3 #8)

Order Reference: [FT]. . .$23.95 + shipping

APPLE DISK LIBRARIAN lets you get control of those big DOS 3.3 disk libraries. Handles up to 500 disks with sorting, editing, and search. (NE 4)

FAST GO-TO PROCESSOR is a machine language shortcut to Applesoft subroutines! Up to 100% speed improvement with GOTO and GOSUB commands. (NE 4)

'PILLAR MUNCH is a fast action arcade game. Watch your 'pillar grow! Munch enough, and your 'pillar turns into a butterfly. (NE 4)

Order Reference: [LB]. . .$29.95 + shipping

APPLE IRS is a tax estimating program for evaluating alternative income, investment and tax strategies. (V4 #1)

TINY TYPER is a short, powerful text editor with append, erase, compact, format, and editing of text files. (V4 #1)

COMPACT SORT does fast machine language sorting of Applesoft arrays in ascending/descending sequences. (NE 4)

LINE EDIT uses ROM techniques for full width Applesoft screen listings. (V4 #1)

Order Reference: [IR]. . .$23.95 + shipping

THE NIBBLE INVESTOR is a comprehensive tracking, analysis, reporting and graphics charting system for your stocks and investments. A best-seller with five different reports. (NE 4)

DISK MAP SYSTEM is a disk analysis program that prints and analyzes your DOS 3.3 disk files, their location and use of disk space. (V4 #2)

SPACE ROVER is an arcade game demonstrating the use of graphics scrolling of landscapes. (V4 #2)

Order Reference: [IV]. . .$29.95 + shipping

CHAMP-R is an indexed reporting system for checks and accounts developed with the CHAMP, Checkbook Management System. (V4 #2)

FORMULA I is a Hi-Res racing game with five different track patterns, barriers and speeds. (NE 4)

FILE NAME MOVER is a DOS 3.3 disk utility that lets you switch/exchange catalog names to put them in the proper sequence. (V4 #2)

Order Reference: [FO]. . .$23.95 + shipping

THE NIBBLE DESIGNER AND ILLUSTRATOR are companion programs for creating, editing, and displaying complex graphics screens. They let you scale, edit, rotate, and even redraw your shapes in medium resolution graphics and then retrieve and control them in building your own design. (NE 4)

DISK ZAP lets you read, examine, edit, and write your disk sectors directly. Great for exploring disk storing techniques and repairing damaged sectors. (NE 4)

NIBBLE INFERNO is a Hi-Res arcade game in which you rescue children from a burning orphanage. (V4 #3)

Order Reference: [DS]. . .$29.95 + shipping

POINTERS is a set of Applesoft utilities to recover from resets, FP commands, and split and overlay programs. (V4 #3)

SORT FIVE is a case study in five different Applesoft sorting techniques. (V4 #3)

SELF-MODIFYING APPLESOFT lets you change Applesoft programs while they're running without losing variables. (V4 #3)

INTERVAL TIMER lets you time up to 25 events with control signals in machine language. (NE 4)

HIDE/UNHIDE lets you store and retrieve your DOS 3.3 programs into a RAM memory expansion card. (V4 #3)

Order Reference: [PT]. . .$24.95

NIBBLE RAM MANAGER is a powerful DOS 3.3 utility that puts up to ten programs into a 16K RAM card for instant access at the touch of a key. (NE 4)

EXPENSE CALC is an easy-to-use spreadsheet program for weekly expense reporting with full scrolling and calculations. (NE 4)

SEARCH AND REPLACE automatically scans and globally or selectively replaces strings and variables in your Applesoft programs. (NE 3)

Order Reference: [RA]...$29.95 + shipping

FLY AWAY lets you use a puffer gun to puff feathers, balloons, and other objects before they touch the ground. Arcade action! (NE 4)

AUTOSCREEN lets you create a "magic cursor" bar in your Applesoft programs for menu item selection under arrow key control. (V4 #4)

BADMINTON is a Hi-Res graphics simulation of tournament badminton. (V4 #4)

PRETTY LISTER formats your Applesoft listings to indent multi-statement lines for that "structured programming look." (V4 #4)

Order Reference: [FL]...$23.95 + shipping

THE NIBBLE ORACLE helps analyze decision alternatives based on your priorities and weighted values for each choice. A great executive decision tool. (NE 4)

AMPER FREE SPLITTER lets you wrap Applesoft programs around protected memory areas. (NE 4)

DOTS-ELLO is a Hi-Res graphics "connect-the-dots" game. (V4 #5)

Order Reference: [OR]...$29.95 + shipping

RAM PAD is an electronic notepad that saves up to four pages of text in a 16K RAM card. (V4 #5)

SPELLING TUTOR lets you specify your spelling list and then reinforces learning by taking letters away from each word until you type it from memory. (V4 #5)

TIME TEACHER uses a Hi-Res clock to move the big and little hand to match the time target. (V4 #5)

MATH MONSTER combines Hi-Res graphics with multiplication/division problem solving. Try to solve the problem before the Math Monster gobbles you up. (NE 4)

NEWSAVER DOS recovers Applesoft programs lost from accidental FP's and NEW's. (V4 #5)

Order Reference: [RP]...$23.95 + shipping

THE NIBBLE PROGRAMMER lets you write structured Applesoft programs without line numbers and adds the powerful new commands LOOP and WHILE. Now you can name your subroutines with terms like GOSUB TOTALS for easy recall and self documentation. The PROGRAMMER then writes your program for you! (NE 4)

MAGAZINE ARTICLE TRACKING SYSTEM indexes, sorts, and finds key articles in your library. (NE 4)

THE NIBBLER is one of the best arcade games we've ever published. *Nibble*'s version of PAC-MAN. (NE 4)

Order Reference: [PG]...$29.95 + shipping

WORD FIND presents puzzles in which you find words embedded in letter mazes. (V4 #6)

SLOT FIND automatically examines slots and identifies your Apple configuration. (V4 #6)

SIMPLE SEARCH is a fast machine language search for Applesoft string array matches. (NE 3)

TEXT FORMATTER prevents word wraparound of your printed text. Provisions for breaking between words in 30, 40, 70, and 130 column formats. (V4 #6)

BOUNCING NUMBERS is a fun Hi-Res number recognition program for preschoolers. (V4 #6)

Order Reference: [WD]...$23.95 + shipping

TUNES is a comprehensive music system that lets you create songs and sound effects. Sharps, flats, duration, staccato, pauses, and all of the other piano/organ effects are there at your fingertips. (NE 4)

APPLESOFT PROGRAMMING HELPER eliminates 50-80% of your program errors before you RUN your program by checking your syntax in advance. (NE 4)

DIGGER lives beneath the sands of Mars. Help him outwit his adversaries in this Hi-Res arcade game. (NE 4)

Order Reference: [TN]...$29.95 + shipping

FILL-REVERSE adds spectacular effects to your Hi-Res graphics displays. (V4 #7)

FUNCTION GRAPHER displays two arithmetic functions on the Hi-Res screen. (V4 #7)

MULTICOPY makes up to seven copies of a disk with a single read. (V4 #7)

ENCODE-DECODE encrypts or decrypts Applesoft and binary files for program security using a 5-character key. (V4 #7)

Order Reference: [FR]...$23.95 + shipping

RAM DISK makes your RAM card a pseudo disk for access to DOS 3.3 data and programs. Later you can save pseudo disk files to diskette for permanent storage. (NE 4)

PERSONAL APPOINTMENT CALENDAR keeps track of a year's worth of appointments with fast editing and printing. (NE 4)

APPLE TALKER lets your Apple speak using your library of words or sounds. (NE 4)

Order Reference: [RD]...$29.95 + shipping

MUNCHIES lets you outguess the number of dots your computer will munch. (V4 #8)

AMPERPACKER saves space by packing text files before saving them to disk. (V4 #8)

APPLE TRIG plots your favorite trig functions in Hi-Res graphics. (V4 #8)

GRAPHICS 3-D explores three-dimensional graphics and hidden line removal. (V4 #8)

Order Reference: [MU]...$23.95 + shipping

NIBBLE DINER is a personal diet planner that not only provides you with nutritional analysis of foods and entire recipes but also helps keep track of your eating habits. The disk includes a database of over four hundred foods. (NE 5)

DOS EXTENDER makes your favorite machine language routine a permanent part of DOS 3.3. (V5 #1)

ARTILLERY DUEL is a Hi-Res shootout over the mountain tops. (V5, #1)

Order Reference: [DI]...$29.95 + shipping

MX-80 PLOTTER makes your Epson printer into a high resolution plotter. (V5, #1)

SIR is a semi-ultimate input anything routine for entering commas, colons and other characters rejected by Applesoft. (V5, #1)

FAST FILE SEARCH speeds DOS 3.3 random access file searches by up to 20 times. (V5, #1)

BLACK BOX uses reflection and absorption of atoms to find the hidden target. (V5, #1)

Order Reference: [MX]...$23.95 + shipping

APPLESOFT EXPANDER adds a library of routines for your DOS 3.3 Applesoft programs featuring: Hi-Res graphics text, hexadecimal PEEK/POKE and computed GOSUB/GOTO with the RAM card. (NE 5)

NIBBLE GARAGE improves your car care by tracking maintenance and generating repair reminders. (NE 5)

SPEED MAZE challenges you to nagivate a random maze at top speed. (NE 5)

Order Reference: [AE]...$29.95 + shipping

FLASHCARD uses Hi-Res graphics for map drill with record keeping and quizzing. (V5 #2)

HI-RES CHARACTER DRAWING makes text displays on the Hi-Res screen easy and quick. (V5 #2)

SAFE CRACKER challenges you to find the combination and open the safe before the cops arrive. (V5 #2)

FANCY PICTURE LOADER provides special effects for loading Hi-Res graphics. (V5 #2)

CUSTOM CATALOG lets you modify DOS 3.3 for custom file type symbols, volume headers, and file names. (V5 #2)

Order Reference: [FC]...$24.95 + shipping

THE NIBBLE BROKER tracks your stocks and graphically displays your graphics with easy data entry and a variety of reports. (NE 5)

QUICKSORT II is a machine language routine for sorting Applesoft one- and two-dimensional strings and numeric arrays on up to nine keys. (NE 5)

STORM WARNING lets you battle the forces of a Midwestern storm to rescue town people and rebuild their houses. (V5 #3)

Order Reference: [BR]...$29.95 + shipping

nibble® software

ATTACK OF THE KILLER SHAPE TABLES challenges you to zap the killer shapes before they cut your lifelines. (V5 #3)

GRAPHICS TOOLBOX is a collection of graphics utilities for swapping Hi-Res pages by superimposing them or inverting them. (V5 #3)

COMPLETE CATALOG lets you examine and recover deleted DOS 3.3 file names. (V5 #3)

SCREEN EDIT 80 lets you create 80-column text screens for display from your programs (requires an Apple //e, 64K). (NE 5)

Order Reference: [KS]...$23.95 + shipping

THE BASIC ASSEMBLER is an editor/assembler package that is written in Applesoft BASIC. Use it to produce machine language programs quickly and easily. (NE 5)

NIBBLE TV GUIDE keeps track of your TV program events and displays schedule conflicts. (NE 5)

SPINBALL provides arcade action with multiple balls, spinners and barriers. (V5 #4)

Order Reference: [BA]...$29.95 + shipping

MONSTER HUNT challenges you to find the monster before he eats you up. (V5 #4)

AUTOMATIC FUNCTION PLOTTER provides auto/manual scaling and graph overlays for plotting your favorite map functions. (V5 #4)

LIFTOFF lets you save an area of the Hi-Res screen as a shape table. (V5 #4)

APPLESOFT OVERLAYS manages your memory as you swap DOS 3.3 Applesoft program sections in and out of memory from disk. (V5 #4)

Order Reference: [MH]...$23.95 + shipping

NIBBLE KEYSOFT lets Applesoft and DOS 3.3 commands be printed in upper or lower case in the full ASCII character set. (NE 5)

PAYCHECK tracks your income and paycheck deductions for screen or printed reports. (V5 #5)

CASTLE RICHE challenges you to find the fortune hidden in the adventure castle using text and Hi-Res graphics clues. (V5 #5)

Order Reference: [KY]...$29.95 + shipping

MATH MARATHON provides drills in the four basic operations with graphics figuring right on the screen. (V5 #5)

HIDDEN LINES creates 3-D Hi-Res graphics on the Hi-Res screen. (V5 #5)

DOUBLE HI-RES I demonstrates the principles of drawing on the double Hi-Res screen. It requires Apple //e with extended 80-column card or Apple //c. (V5 #5)

ANT RACE is a "text graphics racing game" — requires Apple //e with 80-column card or //c. (V5 #5)

AUTOMATIC CAPITALIZER prints properly capitalized output to your printer. (V5 #5)

Order Reference: [MT]...$24.95 + shipping

NIBBLE FILE CABINET is a flexible database management system for creating variable length records with key sorts and binary tree data storage. A *Nibble* best-seller! (NE 5)

DISK CUSTOMIZER for DOS 3.3 provides one step formatting, volume header customization, and user selected HELLO program type. (V5 #6)

CLAM BAKE challenges you to outrun the jellyfish and eat all the diatoms to win in arcade action. (NE 5)

Order Reference: [CB]...$29.95 + shipping

RPN CALCULATOR includes a visible "stack," error trapping, and help commands for a sophisticated graphics calculator. (V5 #6)

LITTLE ORGAN APPLE lets you play music from the keyboard with Hi-Res graphics. (V5 #6)

DISK MAP II shows where your DOS 3.3 files are located and how your disk is structured. (V5 #6)

KEYBOARD INTERRUPT dumps your text screen to a printer on command. (V5 #6)

Order Reference: [CL]...$23.95 + shipping

APPLESOFT GLOBAL EDITOR provides powerful search and replace editing of your programs over a selected line range. (NE 5)

NIBBLE COLOR BILLBOARD turns your Apple into an electronic billboard for the display of colorful attention-getting messages. (V5 #7)

BRIDGE BIDDER pits you against your Apple in sharpening your bridge playing skills in Hi-Res graphics. (V5 #7)

Order Reference: [AG]...$29.95 + shipping

PEARSON PRODUCT MOMENT CORRELATIONS calculates a complete correlation matrix from your statistical data. (V5 #7)

COUNTING QUIZ helps your preschoolers learn their numbers in Lo-Res and Hi-Res graphics. (V5 #7)

APPLESOFT VARIABLE DUMP shows a complete list of your variables and their values at any time during program execution. (NE 5)

DISK CERTIFIER marks all "bad spots" on either side of your DOS 3.3 diskette. (V5 #7)

Order Reference: [PP]...$23.95 + shipping

SOUND SYNTHESIZER is a sound creation utility that lets you "draw" your sound effects on the Hi-Res screen. Sound effects can be stored in libraries for later use by your own programs. (NE 5)

LIST MASTER formats your Applesoft programs for easy-to-read structured listings. (NE 5)

NIBBLE BEACH HEAD challenges you to lead your troops across the beach head to win in arcade graphics. (V5 #8)

Order Reference: [SS]...$29.95 + shipping

REGRESS provides video and printed regression analysis. (V5 #8)

LITTLE FINGERS helps young typists learn the keyboard. Designed for small hands. (V5 #8)

APPLESOFT ANTI-LINE CRUNCHER expands multi-statement Applesoft programs to give each statement its own line number. (NE 5)

MON-E runs under DOS 3.3 and provides a separate monitor in extended 80-column memory. It lets you switch back and forth to check the current status of your programs—requires a //e with extended 80-column card or //c. (V5 #8)

Order Reference: [RG]...$23.95 + shipping

NIBBLE GRADE BOOK lets you track and compute grades for up to ten classes of 80 students each. Record, analyze, and assign grades with ease. (NE 5)

RAM DISK 64 creates 170-sector pseudo disks on your 128K //e or //c for fast access with normal DOS 3.3 commands. Requires Apple //e with extended 80-column card or Apple //c. (NE 5)

MATHEMAGICIAN is an adventure game that combines education with fun in building math skills. (V5 #9)

Order Reference: [GB]...$29.95 + shipping

COMPASS QUIZ offers drill, practice, and progress tracking for learning the principle directions of the compass in Hi-Res graphics. (V5 #9)

TADPOLE ALPHABET lets your preschooler learn the alphabet in an arcade style game. (V5 #9)

PROMENU lets you select system time and date and explore the ProDOS directory tree with a series of menus — 64K Applesoft and ProDOS required; must be converted before use. (V5 #9)

SOLID STATE SLIDE SHOW stores Hi-Res pictures in your RAM card (or upper 16K on the //e) for superfast display. (V5 #9)

Order Reference: [CQ]...$23.95 + shipping

HI-RES HOUDINI is a powerful machine language graphics utility for creating special graphics effects including shift, merge, swap, and inverting both Hi-Res screens. (NE 5)

POSTMASTER keeps records for up to 100 labels which can be sorted and custom printed with special options that you select. (V5 #10)

3-D FOUR SCORE is a three-dimensional Tic Tac Toe in Lo-Res graphics. (V5 #10)

Order Reference: [HH]...$29.95 + shipping

DHR PALETTE is a graphics utility for designing double Hi-Res graphics screens and saving them to disk — requires a //e with extended 80-column card or //c. (V5 #10)

FATHER'S FIRST PROGRAM is a collection of four programs in graphics designed with children in mind. (V5 #10)

AUTOLOG keeps track of your sessions on the computer using "log on" and "log off" reporting. (V5 #10)

HEX PEEKS, POKES, AND CALLS allows you to

execute these commands from Applesoft using hexadecimal notation. (V5 #10)

Order Reference: [HP]...$23.95 + shipping

COUPMAN will help you keep track of your grocery coupons for greater savings. Analyzes and prints reports based on expiration date, coupon category, and more. (NE 5)

APPLESOFT BUG CHASER helps you eliminate program bugs by continuously displaying the value of a selected variable during program execution. (NE 5)

GOLF PRO lets you design your own golf course and play the holes in Hi-Res graphics. (V5 #11)

Order Reference: [CP]...$29.95 + shipping

CROSSWIND is a dramatic simulation of a football kick with control over velocity, angle, and wind direction in Hi-Res graphics. (V5 #11)

COMPOSITE BIORHYTHMS shows your biocycles individually and combined in Hi-Res graphics. (V5 #11)

MACHINE CODE EDITOR disassembles machine code as you enter it for instant feedback and comparison to the original. (V5 #11)

Order Reference: [CW]...$23.95 + shipping

STOCK ANALYST automatically evaluates your portfolio, including the status and history of individual stocks and your entire holdings. (NE 5)

AMPERGO is a powerful ampersand utility that lets you write Applesoft programs using labels for GOTO and GOSUB statements. (NE 5)

OBELISK challenges you to pilot your rover vehicle to the mother ship, avoiding death rays from alien ships and obelisks. (V5 #12)

Order Reference: [AN]...$29.95 + shipping

MATH CONCENTRATION reveals "concentration panels" as a reward for solving math problems in Hi-Res graphics. (NE 5)

MORTGAGE CALCULATOR displays your monthly payment and total payments when a loan amount, interest rate, and term are entered. (V5 #12)

DOUBLE HI-RES GRAPHICS ROUTINES include the driver routines and sample programs presented in the Graphics Workshop — 128K required on the //e or //c. (V5 #12)

RAM RESERVATION CENTER is a ProDOS utility for reserving or freeing areas of RAM so that ProDOS will respect the new boundaries — requires ProDOS and at least 64K. (V5 #12)

Order Reference: [MM]...$23.95 + shipping

NIBBLE ARCHITECT is a powerful program for planning your room design. Scaled, Hi-Res graphics objects can be easily moved, changed, copied, printed, and saved to disk for professional quality results. (V6 #1)

POWERKEY lets you define single keyboard macros for simple entry of DOS 3.3 commands and program statements. Dramatically reduces program typing. (V6 #1)

CLAUSTROPHOBIA pits you against aliens from the sky who try to trap your gunner. Bring them down in neat columns at the edge of the screen to win. (V6 #1)

Order Reference: [NA]...$29.95 + shipping

SYSTEM SOLVER will solve any set of up to 70 linear equations with up to 70 unknowns. (V6 #1)

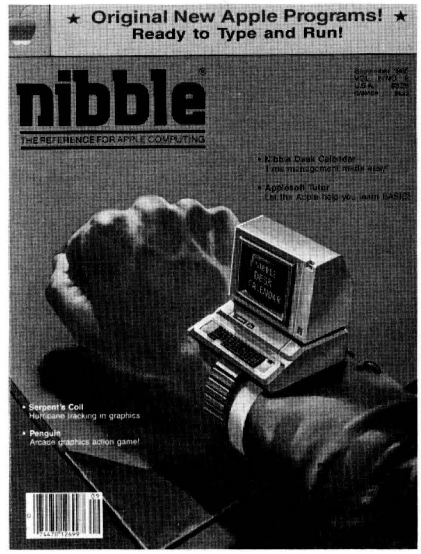
CALORIE COUNTER helps track your calorie consumption while it builds a database on disk as you add new foods to your diet. (V6 #1)

CATALOG PLUS lets you find and catalog DOS 3.3 disks by file type and initial character. (V6 #1)

APPLESOFT EXECUTION MONITOR graphs a chart of your program's execution showing the program statements that are executed most often. A powerful tool for optimizing Applesoft programs. (V6 #1)

Order Reference: [SY]...$23.95 + shipping

SPOOLER keeps your printer busy while you keep computing. Uses the 16K RAM card area to store printer output as you work under DOS 3.3. (V6 #2)



NIBBLE PAGE EDITOR lets you create up to 8 pages of information with cursor oriented screen editing, printing, or disk storage. (V6 #2)

THE NIBBLE 500 challenges you to race for glory through a curving track with random obstacles and a pit stop. (V6 #2)

Order Reference: [SP]...$29.95 + shipping

PERPETUAL CALENDAR will print a full page calendar for any year since 1753. (V6 #2)

DOUBLE HI-RES ROUTINES include the final routines for vertical shifting in the double Hi-Res driver, including a demonstration program and the entire DHR driver — requires 128K //e or //c. (V6 #2)

DOS TRICKS lets you obtain disk boot statistics, read/write statistics, and create a "scratch pad" text file on DOS 3.3 disk for quick reference information. (V6 #2)

DISK LOCK helps you avoid writing over valuable data by fooling DOS 3.3 into thinking that your disk is full. (V6 #2)

Order Reference: [PC]...$23.95 + shipping

NIBBLE MEDIC will help you keep track of your family's medical expenses and prescription history with a variety of editing and report formats. (V6 #3)

TINY COMPILER is an Applesoft post-processor that speeds your program's execution up to 20%. (V6 #3)

TRIVIA MASTER challenges you to beat the clock at answering trivia questions. The disk includes a file of 200 questions to get going. (V6 #3)

Order Reference: [MD]...$29.95 + shipping

LUCK converts all your upper and lower case program text to upper case for easy formatting of listings. (V6 #3)

TEXT VIEWER scrolls quickly through your text files without having to boot your word processor. (V6 #3)

PRODOS CRYPTOGRAPHY disassembles ProDOS's message printer and includes a utility printer to build and display your own messages. (V6 #3)

AMPERDHR is a library of graphics routines to simulate normal Hi-Res commands in double Hi-Res graphics — requires 128K. (V6 #3)

SCREEN SPINNER builds eye-catching screen borders by spinning your message around the perimeter of the display. (V6 #3)

Order Reference: [LK]...$24.95 + shipping

APPLE HIGHWAYS will find the best route between 170 major U.S. cities and print both the route and the mileage between junctions. (V6 #4)

NIBBLE GAS MISER tracks your gas mileage and graphs the results in Hi-Res graphics. (V6 #4)

IDOL OF MONTEREY is a text adventure game in which you battle the monsters of the forest to regain the precious idol. (V6 #4)

Order Reference: [HI]...$29.95 + shipping

APPLE TIME TUTOR uses the Hi-Res screen with several clock formats and levels of difficulty for youngsters of different ages. (V6 #4)

BLOCK SHAPE ANIMATION IX includes a series of routines for single-byte block shape animation. (V6 #4)

BEEP CUSTOMIZER lets you modify Apple's tone and include the changes in your own programs. (V6 #4)

PRODOS DATE AND TIME allows you to set the date and time without a clock card within ProDOS. (V6 #4)

INPUT AND EDIT uses the ampersand to allow editing within an input field. (V6 #4)

Order Reference: [TM]...$24.95 + shipping

nibble software

APPLESOFT SUPERCHARGER is a set of machine language utilities that use ampersand commands to format output, control user input, chain programs with common variables, and much more. (V6 #5)

BANNER BOSS creates custom banners in two letter sizes with underlining and emphasized print features. (V6 #5)

ADVENTURE CONSTRUCTION SET lets you design and play your own adventure games. (V6 #5)

Order Reference: [BB]. . .$29.95 + shipping

UNIVERSAL METRIC CONVERTER converts English and metric units with menu-driven ease. (V6 #5)

AUXILIARY MEMORY EXAMINER is an ampersand-driven utility that lets you examine any portion of the 128K of RAM in an Apple //c or extended //e. (V6 #5)

BASIC ASSEMBLER ENHANCEMENTS improves *Nibble's* BASIC Assembler with powerful printing and editing features. (V6 #5)

PAGE lets you draw Hi-Res pictures and save them on disk as screens or as program lines that you can merge with your favorite Applesoft routine. (V6 #5)

Order Reference: [UM]. . .$23.95 + shipping

CHART MANAGER creates sophisticated Hi-Res scatter plots, bar graphs and pie charts from data you key in or read from disk. (V6 #6)

DOS DESIGNER is a menu-driven system that allows you to change DOS 3.3 commands, create custom error messages, alter the disk volume heading, and more. (V6 #6)

ANTS! is a Hi-Res action game that pits you against hordes of giant, invading ants bent on destruction! (V6 #6)

Order Reference: [CM]. . .$29.95 + shipping

BLOCK SHAPE ANIMATION X is an ampersand command interpreter for block shape animation that lets you acccess commonly used routines without using their CALL addresses. (V6 #6)

CHICKEN LITTLE'S TYPING GAME teaches you the basics of touch typing. To score you must type the letters that fall from the top of the screen. (V6 #6)

VIGILANT FID lets you store the FID program in the RAM card area of memory and access it with a simple command for speed file manipulation under DOS 3.3 (V6 #6)

PRODOS TYPE COMMAND adds a convenient TYPE command to ProDOS so you can dump an ASCII version of any file to any output device in immediate and deferred modes. (V6 #6)

PROGRAMMING THE APPLEMOUSE II demonstrates mouse programming techniques with Applesoft and machine language. Create line drawings on the Lo-Res screen and determine the mouse's coordinates with these two sample programs. (V6 #6)

Order Reference: [BS]. . .$23.95 + shipping

NAVIGATOR makes program documentation a breeze! It formats Applesoft listings and prints flowcharts complete with graphics symbols. (V6 #7)

NIBBLE MAESTRO transforms the Apple keyboard into a four-octave organ. Notes appear on a staff as you play, and you can save tunes to disk and edit or play them later. (V6 #7)

LIGHTNING COPY copies disks quickly with a two-drive system under DOS 3.3 You control the error handling for fast, foolproof copying, even with damaged disks. (V6 #7)

Order Reference: [NV]. . .$29.95 + shipping

IRA COMPARISON CHART calculates the future value of an Individual Retirement Account (IRA) and compares it to your return on another investment. It even allows you to adjust for inflation. (V6 #7)

ASSEMBLY II includes two machine language programs. One produces music from the Apple keyboard; the other converts between the binary, decimal and hexadecimal number systems. (V6 #7)

EYE OPENERS creates an opening iris effect for dramatic transitions from one Hi-Res display to the next. (V6 #7)

SUPER REM REMOVER removes REMs from Applesoft programs for faster program execution and more compact code. Run the speedy, REM-less version of your program, and save a fully documented program listing for reference. (V6 #7)

Order Reference: [IC]. . .$23.95 + shipping

EXECUTIVE CARD FILE is an electronic index card file that features easy data entry and editing, speedy sorting, and disk-based storage. Each index card can hold nearly 500 characters, and files can be divided into seven sections with 35 index cards in each. (V6 #8)

PRODOS-DOS 3.3 DOUBLEBOOT lets you boot ProDOS or DOS 3.3 from a single disk. (V6 #8)

NIBBLE BLACKJACK gives you the excitement of Las Vegas style blackjack on the Hi-Res screen. The game includes splitting on pairs, doubling down, and five-card and six-card specials. (V6 #8)

Order Reference: [EX]. . .$29.95 + shipping

DYNAMIC FUNCTIONS offers a pair of ampersand routines to convert strings to functions and then evaluate the converted string. Two demonstration programs are included.

INVERSE KEYWORDS highlights every occurrence of a selected keyword in an Applesoft listing for easier program debugging. The program runs under DOS 3.3 with 64K RAM. (V6 #8)

DOUBLE HI-RES GRAPHICS lets you create crisp Hi-Res displays with double the normal horizontal resolution using the Apple II Plus. The bonus: you don't need extra hardware! (V6 #8)

APPLESOFT SCREEN CREATOR converts your text screens to Applesoft code. An editor allows normal, inverse, and flashing characters. (V6 #8)

Order Reference: [DY]. . .$24.95 + shipping

NIBBLE DESK CALENDAR is a quick-reference desk calendar that lets you highlight dates and print out a list of the corresponding events. Menu bar selection makes entering, changing or deleting events a snap! (V6 #9)

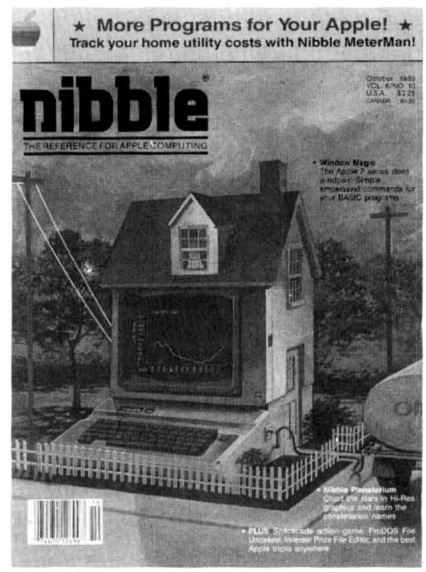
APPLESOFT TUTOR sharpens your debugging skills with a system of detailed error messages for a hundred different errors. (V6 #9)

PENGUIN is an arcade-style strategy game with increasing skill levels. Help Sammy the Penguin fend off the invaders to his little island! Sliding ice blocks are your weapons. (V6 #9)

Order Reference: [DC]. . .$29.95 + shipping

ASCII.DUMP (Nibbling at Assembly Language) lets you stop, pause, restart and step through a hex dump at your own pace. It also prints the hexadecimal values of numbers stored in memory and their ASCII equivalents. (V6 #9)

THE SERPENT'S COIL is a Hi-Res graphic hurricane tracking program for the Caribbean and Gulf of Mexico. Features include easy data entry and update, single or multiple storm plots, scaling and labeling of charts. (V6 #9)



HEX SEQUENCE FINDER searches for the sequence of hex code you specify and locates each occurrence — fast! Use it to debug your machine language programs; just enter the starting page, ending page and the search sequence. (V6 #9)

SUBROUTINE STORE keeps your favorite subroutines on file for future use. Descriptive REM statements tell you exactly what each subroutine does. List the subroutine you want, renumber it, add it to another program and save the result. (V6 #9)

Order Reference: [AD]. . .$24.95 (Introductory price $17.95 expires 11/30/85) + shipping

---

NIBBLE METERMAN tabulates your utility payments and graphs them for easy cost analysis. (V6 #10)

WINDOW MAGIC creates Mac-like windows that you can move anywhere on the Apple II Plus, //e or //c screen. Create any size window, overlap windows and direct I/O to them. (V6 #10)

nibble® software

SPACECADE challenges you to dodge the meteors as they fall faster and faster. It takes speed and agility to survive Spacecade! (V6 #10)

Order Reference: [WM]. . .$29.95 + shipping.

NIBBLE PLANETARIUM is a Hi-Res star charting system with the features of a real planetarium. Display and print the sky from any location in the Northern Hemisphere and at any time. A great learning tool for amateur astronomers! (V6 #10)

INVESTOR PRICE FILE EDITOR is an enhancement to Nibble Investor. It sorts, deletes and quickly adds records to the Investor's price files. (You need Nibble Investor, Order Reference IV, to use this program.) (V6 #10)

TEXT FILE UTILITY creates and edits text files easily with features that include line numbering, character insert and delete, and a range of disk options. (V6 #10)

RECOVERING DELETED PRODOS FILES instantly exhumes deleted files under ProDOS. (V6 #10)

Order Reference: [PN]. . .$23.95 + shipping.

SUPER SHOPPER is a database that files a master shopping list of up to 1,600 separate items and prints out custom shopping lists to bring to the store.

ATLANTIC CONVOY pits your navy against the enemy in an all-out struggle for the Atlantic domain. You need strategy and skill to maneuver your fleet of thirteen vessels with various defense, move and attack capabilities.

SUBROUTINE MASTER adds real subroutine handling to Applesoft. Features include references to subroutines by name, two-way parameter passing, recursion and local variables.

Order Reference: [SU]. . .$29.95 (Introductory price $19.95 expires 1/31/86) + shipping.

DIRECTION FIELDS plots direction fields using the mathematical function you supply.

HI-RES CRUNCH-DECRUNCH compacts Hi-Res pictures for efficient disk storage, then restores the data to its original size for Hi-Res display.

APPLESOFT CROSS REFERENCE prints out four reports that list your programs' references to line numbers, variable names, literal values and constants. An invaluable debugging aid!

PROSIR streamlines input under ProDOS. An "input anything" routine allows you to restrict the length of a field and a phantom numeric keypad speeds numeric input.

NIBBLING AT ASSEMBLY LANGUAGE IV includes two programs that create sounds and convert hexadecimal numbers to decimal as they demonstrate the 65C02 instructions that load, store, compare, transfer and branch.

Order Reference: [DF]. . .$24.95 (Introductory price $17.95 expires 1/31/86) + shipping.

NIBBLE INVESTMENT ADVISER calculates return on investment and tracks dividends for up to 96 different investments. Use it to find out how well your investments are performing right now!

PRODOS COPY COMMAND adds a COPY command to ProDOS for efficient file manipulation. Transfer files from within a BASIC program or an EXEC'able text file.

TITAN CRONUS challenges you to battle the ancient gods in fast-paced arcade action. Fend off waves of attackers to win.

Order Reference: [IA]. . .$29.95 (Introductory price $19.95 expires 2/28/86) + shipping

---

BASIC DISASSEMBLER translates machine language code into symbolic source code that your assembler can read. Use it to modify, correct or enhance programs for which you have no source code.

NIBBLE DUET jazzes up your Apple with two-voice sound. Create sophisticated sound effects to complement your programs.

JUST WAITING demonstrates the use of the WAIT command to create a pause in your program while displaying no cursor or a custom cursor.

ANATOMY OF A TWO-LINER is a collection of thirteen different games — each written in just two lines. Challenge your dexterity and speed with Dodgeball, Serpent, Find and Maze.

HAPPY BIRTHDAY plays "Happy Birthday" and displays the words in time with the song, complete with the birthday person's name.

Order Reference: [BD]. . .$24.95 (Introductory price $17.95 expires 2/28/86) + shipping

This month you can find nibbLEADS on pages 157-159

nibble SOFTWARE ORDERING COUPON

INTRODUCTORY PRICE OFFER

For a 90-day period following the publication of each issue, NIBBLE will offer the major programs from that issue on a DISKETTE for a nominal "Introductory Price."

Following the 90-day introduction, the program will be offered with Reprints of the original NIBBLE articles which documented them. In addition, where specifically noted, the programs will be provided with significant enhancements which have been added to the diskettes.

During the introduction period, disks are shipped without accompanying documentation under the assumption that the buyer has the issue in which the programs appeared.

NC ☐ **Turtle BASIC/Nibble Calculator/Tank Combat**...Volume 7, Issue #1. Introductory price **$19.95** expires 3/31/86.

IA ☐ **Nibble Investment Adviser/ProDOS Copy Command/Titan Cronus**...Volume 6, Issue #12. Introductory price **$19.95** expires 2/28/86.

SU ☐ **Super Shopper/Subroutine Master/Atlantic Convoy**...Volume 6, Issue #11. Introductory price **$19.95** expires 1/31/86. New price: $29.95.

LP ☐ **Nibble Light Pen/ProDOS Directory List/ ProCursor/DISPLAY/Nibbling at Assembly Language V**...Volume 7, Issue #1. Introductory price **$17.95** expires 3/31/86.

BD ☐ **BASIC Disassembler/Nibble Duet/Just Waiting/Anatomy of a Two-Liner/Happy Birthday**...Volume 6, Issue #12. Introductory price **$17.95** expires 2/28/86.

DF ☐ **Direction Fields/Hi-Res Crunch-Decrunch/ Applesoft Cross Reference/ProSIR/Nibbling at Assembly Language IV**...Volume 6, Issue #11. Introductory price **$17.95** expires 1/31/86. New price: $24.95.

The disks below are listed according to the first program on the disk. Each disk contains multiple programs; for complete descriptions see the catalog preceding this order form.

TR ☐ **Apple TRAC Plus** V1/#1. **$24.95**
SA ☐ **Star Attack** V1/#2. **$19.95**
TG ☐ **Apple Tough** V1/#3. **$24.95**
FW ☐ **F.A.S.T** V1/#4. **$24.95**
P2 ☐ **PIP II** V1/#6. **$29.95**
AC ☐ **Apple A.I.M.** V1/#7. **$28.95**
WO ☐ **Will 'O The Wisp** V1/#8. **$19.95**
CH ☐ **Apple Champ** V2/#1. **$28.95**
MR ☐ **Music Retrieval** V2/#2. **$19.95**
PB ☐ **Apple Paintbox** V2/#3. **$24.95**
ST ☐ **Apple Star System** V2/#4. **$29.95**
DM ☐ **Apple Disk Master** V2/#5. **$29.95**
MA ☐ **Apple M.A.P.S.** V2/#6. **$29.95**
DA ☐ **Apple Dart** V2/#7. **$29.95**
FI ☐ **Financier III** V2/#7. **$19.95**
TB ☐ **TRAC "B"** V2/#8. **$29.95**
HX ☐ **Hexpad** V2/#8. **$19.95**

TU ☐ **Turtle Graphics** V3/#1. **$29.95**
TT ☐ **Apple Turtle Graphics Tutor** V3/#1. **$23.95**
SC ☐ **Scramble** V3/#1. **$19.95**
ML ☐ **Nibble MLE** V3/#2. **$29.95**
SL ☐ **Apple Slugger** V3/#2. **$19.95**
BN ☐ **Apple Bond Manager** V3/#3. **$29.95**
AM ☐ **Amperspeed** V3/#3. **$22.95**
MC ☐ **Micro-Calc** V3/#4. **$29.95**
AB ☐ **Abacus** V3/#4. **$22.95**
AL ☐ **A.L.E.** V3/#5. **$29.95**
DD ☐ **DDT** V3/#5. **$22.95**
AR ☐ **A.R.C.** V3/#6. **$29.95**
BC ☐ **Binary Clock** V3/#6. **$22.95**
AP ☐ **AMP-L-SOFT** V3/#7. **$29.95**
SH ☐ **Shape** V3/#7. **$23.95**
RB ☐ **Recipe Box** V3/#8. **$29.95**
FT ☐ **Fat Graph** V3/#8. **$23.95**
LB ☐ **Disk Librarian** V4/#1. **$29.95**
IR ☐ **I.R.S.** V4/#1. **$23.95**
IV ☐ **Investor** V4/#2. **$29.95**
FO ☐ **CHAMP-R** V4/#2. **$23.95**

DS ☐ **Designer** V4/#3. **$29.95**
PT ☐ **Pointers** V4/#3. **$24.95**
RA ☐ **RAM Manager** V4/#4. **$29.95**
FL ☐ **Fly Away** V4/#4. **$23.95**
OR ☐ **Oracle** V4/#5. **$29.95**
RP ☐ **RAM Pad** V4/#5. **$23.95**
PG ☐ **Nibble Programmer** V4/#6. **$29.95**
WD ☐ **Wordfind** V4/#6. **$23.95**
TN ☐ **T.U.N.E.S** V4/#7. **$29.95**
FR ☐ **Fill-Reverse** V4/#7. **$23.95**
RD ☐ **RAM Disk** V4/#8. **$29.95**
MU ☐ **Munchies** V4/#8. **$23.95**
DI ☐ **Nibble Diner** V5/#1. **$29.95**
MX ☐ **MX-80 Plotter** V5/#1. **$23.95**
AE ☐ **Applesoft Expander** V5/#2. **$29.95**
FC ☐ **Flashcard** V5/#2. **$24.95**
BR ☐ **Nibble Broker** V5/#3. **$29.95**
KS ☐ **Killer Shape Tables** V5/#3. **$23.95**
BA ☐ **Basic Assembler** V5/#4. **$29.95**
MH ☐ **Monster Hunt** V5/#4. **$23.95**
KY ☐ **Keysoft** V5/#5. **$29.95**

MT ☐ **Math Marathon** V5/#5. **$24.95**
CB ☐ **File Cabinet** V5/#6. **$29.95**
CL ☐ **RPN Calculator** V5/#6. **$23.95**
AG ☐ **Applesoft Global Editor** V5/#7. **$29.95**
PP ☐ **PPMC** V5/#7. **$23.95**
SS ☐ **Sound Synthesizer** V5/#8. **$29.95**
RG ☐ **Regress** V5/#8. **$23.95**
GB ☐ **Nibble Grade Book** V5/#9. **$29.95**
CQ ☐ **Compass Quiz** V5/#9. **$23.95**
HH ☐ **Hi-Res Houdini** V5/#10. **$29.95**
HP ☐ **Double Hi-Res Palette** V5/#10. **$23.95**
CP ☐ **Coupman** V5/#11. **$29.95**
CW ☐ **Crosswind** V5/#11. **$23.95**
AN ☐ **Stock Analyst** V5/#12. **$29.95**
MM ☐ **Math Concentration** V5/#12. **$23.95**
NA ☐ **Nibble Architect** V6/#1. **$29.95**
SY ☐ **System Solver** V6/#1. **$23.95**
SP ☐ **Spooler** V6/#2. **$29.95**
PC ☐ **Perpetual Calendar** V6/#2. **$23.95**
MD ☐ **Nibble Medic** V6/#3. **$29.95**
LK ☐ **LUCK** V6/#3. **$24.95**

HI ☐ **Apple Highways** V6/#4. **$29.95**
TM ☐ **Apple Time Tutor** V6/#4. **$24.95**
BB ☐ **Applesoft Supercharger** V6/#5. **$29.95**
UM ☐ **Universal Metric Converter** V6/#5. **$23.95**
CM ☐ **Chart Manager** V6/#6. **$29.95**
BS ☐ **Block Shape Animation X** V6/#6. **$24.95**
NV ☐ **Navigator** V6/#7. **$29.95**
IC ☐ **IRA Comparison Chart** V6/#7. **$23.95**
EX ☐ **Executive Card File** V6/#8. **$29.95**
DY ☐ **Dynamic Functions** V6/#8. **$23.95**
DC ☐ **Nibble Desk Calendar** V6/#9. **$29.95**
AD ☐ **ASCII.DUMP** V6/#9. **$23.95**
WM ☐ **Nibble MeterMan** V6/#10. **$29.95**
PN ☐ **Nibble Planetarium** V6/#10. **$23.95**

Complete product descriptions begin on page 148.

Please send me the products I have checked.

I am paying by:

☐ Check ☐ Money Order ☐ VISA ☐ MasterCard

Card # _____ Exp. _____

Signature _____

(ALL PRICES PLUS SHIPPING)

U.S. Shipments please add $1.50 for shipping and handling. Outside U.S. add $2.50 for the first disk ordered and $1.00 for each additional disk ordered. MA residents add 5% Sales Tax. *NOTE*: Disk updates require return of orginal Nibble diskette. Add shipping and handling.

Payable in U.S. funds only.

Name _____

Address _____

City _____

State _____ Zip _____

Telephone _____

Mail this order form to:

nibble 45 Winthrop St., Concord, MA 01742

nibbLEADS

RATES AND DEADLINES

PAYMENT

1x Rate: Five line minimum at $67.50. $15.00 per line up to 10 lines.
3x Rate: Five line minimum at $62.50. $12.50 per line up to 10 lines.

All additional full or partial lines at $17.50 each.

All ad placement requires prepayment in full with the issue of first insertion.

Regular lines consist of 50 characters/spaces. Bold lines consist of 40 characters/spaces. Please be sure to indicate those sentences you wish to be **printed in bold**. Due to variances of type as published, your ad may vary in line count from your typed copy. Name, address and phone number are not tallied in the count, unless they are centered and/or on separate lines. Please indicate in which category you would like your ad to appear.

DEADLINES

Typed copy and prepayment for all ad insertions must be submitted by the 10th of the second month prior to cover date, e.g. January 10th for the March issue. All ads received after deadline will be held for the next issue. No verbal reservations are accepted. Nibble reserves the right to reject any advertisement deemed unacceptable for its readership.

To run your ad in nibbLEADS send your prepayment and copy, or address further questions to: **Classified Ad Department, nibbLEADS, 45 Winthrop Street, Concord, MA 01742, (617) 371-1660.**

SOFTWARE

INTERNATIONAL COMPATIBLES

Supporting II, II+, FRANKLIN, & ALL COMPATIBLE users, with a monthly newsletter, public domain library & personal attention. Membership $20 a year. Disks: $12 DS/DD, $27 3M SS/DD. Public Domain Samplers, $23 per 7 disk set (3M $32). Sets: Games, Eamon, Utilities, Music, Education, CP/M, Pascal, Business, Graphics, & Religion. Franklin Users Group International, 64 East Wind, Suite L, Tecumseh, MO 65760.

DOWNLOAD CUSTOM CHARACTERS FAST! TO YOUR APPLE DMP OR IMAGEWRITER

Here's the answer you've been looking for. Complete step-by-step instructions, examples, program listings, worksheets. Everything you need to download custom characters. Includes program codes for 5 fonts: Italics, Scientific, Greek, Script, and Astronomy — plus Printer Graphics instructions and Apple printer Utilities. This one really works!

Custom Characters & Graphics Book $19.95

Manual PLUS Menu-Driven DOS 3.3 software $37.50. Add $2.00 for First-Class Postage. Write for info. TIMBERLINE PRESS, INC., P.O. BOX 211326, Denver, CO 80221, (303) 429-4053.

APPLE — IBM BISYNC COMMUNICATION

Transfer files to/from Apple II/IIe & IBM mainframe POWER, JES, etc. + self contained editor, etc. with APPLE-BISYNC 3780 RJE Workstation Emulation. Also 3270 Terminal Emulation. URGEO SOFTWARE, Inc. P.O. Box 305, Cheney, WA 99004 (509) 838-6058.

> > ProImage: an Animation system < <

A menu driven interactive system for adding full screen smooth flowing professional quality animations to BASIC or machine language programs. Easy to learn and use. Specify DOS 3.3 or ProDOS when ordering. Send $34.95 to Interactive Arts, 2715 Porter St., Soquel, CA 95073, (408) 475-7047.

APPLE PASCAL USERS

*MONITOR ($49) Pop-up monitor has calculator, key macros, notepad, print screen, filter, assembly and more. *EDIT ($39), faster and more versatile than system editor, Applewriter or ASE. *STAT ($99) has regressions, ANOVAs, eigenvalues, matrix algebra, principle components, nonparametrics. *UTIL ($39) has program print, disk manager, decompile/debug. dogStar Software, Box 302, Bloomington, IN 47402, 812/333-5616.

GRAPHICS

FLOWCHARTING MADE EASY

Create, edit and print full page flowcharts with **CHARTS UNLIMITED**. Flowchart objects are provided or create your own. Fast graphics and easy to use. FREE brochure. Works on Apple II series. Only $195.00. Money-back guarantee. Call or write Graphware, Inc., PO Box 373, Middletown, OH 45042, (513) 424-6733.

PRODUCE LARGE ORGANIZATION CHARTS

Need more than one screen to draw your organization chart? Then you need CHARTS UNLIMITED. Graphics area equivalent to 12 hi-res screens. Used by Fortune 500 companies. Supports most printers & plotters and the Apple II series. Call or write Graphware, Inc., PO Box 373, Middletown, OH 45042, (513) 424-6733.

II+ CAD

Comprehensive, menu-oriented, graphics program for students and engineers. Isometrics, wire frames, architectural, etc. Rectilinear and polar lines, circles, triangles, rectangles, ellipses (plain, solid, dashed, rotated), arcs, arrows, cursor draw erase/fill, load, save, any size text. With coord. graph pad and Molecule Draw program. For II+, IIe, IIc, III. $29.95 + $2.00 s/h. AZ res. +6%. CADSOFT, P.O. Box 1798, Cottonwood, Arizona 86326.

UTILITIES

The TURBO Assembler — your dream come true: A powerful 6502/65C02 $49.95 assembler, 5 to 100 times faster than anything you've used before, combined with a superb word-processor like editor to make your programming quick, easy, and enjoyable. $5 for demo disk with documentation. Write: TURBOSOFT, 28 Stratford Rd., Montreal, Canada H3X 3C5.

Send for free list of programs. **When**
Uncle Truck isn't working on the farm, he writes utilities and games for us cousins. Cousin Hector, the Marine, tests them, and cousin Harlow, who doesn't talk, documents them. Write to Uncle Truck's Software Tool & Die, PO Box 2437, Fullerton, CA 92633.

HACKERMAN STAN™ AND HIS AMPERSAND BAND™

Create high quality programs, quickly with this combination program-generator and BASIC commands extension (WINDOW's, user-defined INPUT, desk accessory calculator, more). 128K //e, //e enhanced, or //c. $39.95 (+$2 S&H) price includes FREE sub. to CBBS-free software on-line! Check/MO-Coppersoft Solutions, 1512 Atlantic/P.O. Box 404, Hancock, MI 49930. (906) 482-8160.

APPLE DMP & IMAGEWRITER USERS PRINTER PROBLEMS?

Here's the book you've been waiting for. How to program your Apple dot matrix printer. NEW 100-page manual gives all printer commands in Applesoft BASIC. Great for beginners. Detailed instructions, examples, and glossary, plus 18 ready-to-run programs. $9.95 ppd. Money-back guarantee. FREE INFO. write: "PRINTER MANUAL", TIMBERLINE PRESS, INC., P.O. Box 211326, Denver, CO 80221 (303) 429-4053.

SUPPLIES/ACCESSORIES

DISK SALE! —SS/DD 35-track for Apple w/sleeve & label-10/$5.80, bulk-100/$45. Standard SS/DD w/sleeve & label-10/$7.50, bulk-100/$59. DS/DD w/sleeve & label-10/$8.50, bulk-100/$67. PREMIUM QUALITY, LIFETIME WARRANTY! Money back satisfaction guarantee! Min order, $20. Send check or pay by MC/VISA/AE. $3 shipping, + $2 if cod. Unitech, 20 Hurley St., Cambridge, MA 02141, (800) 343-0472, in MA (617) 864-8324.

SAVE ••• DISKETTES ••• SAVE

Disks listed are 5.25 1S/2D. WABASH 14.50/10, 3M 17.00/10, BASF 14.50/10, BONUS 12.00/10, POLAROID 18.50/10. Also available are other disks, ribbons, software, surge protectors, labels etc. 1.50 S/H. KODIAK ASSOCIATES, PO Box 2731, Decatur, AL 35601.

HOME/PERSONAL

HOUSECALL — Medical Encyclopedia

An Apple today keeps the Doctor away. Apple II up. 250K database gives medical advice on 400 diagnosis accessed through symptoms or directly. Always ready for a "second opinion". 3 disks. Special $39.95 + $3 s&h. (CO +3%). Orders: (800) 233-3556 or (303) 773-1237. Rocky Mountain Medical, P.O. Box 4783, Englewood, CO 80155.

Vegetable garden planning program for your home computer. On screen layout and user controlled modifications. Completely customizable vegetable tables. Commodore 64 ($29.95), Apple II+/IIe/IIc ($39.95), IBM PC/PCjr ($45.00). The Gardener's Assistant. Shannon Software Ltd, PO Box #6126, Falls Church, VA 22046, (703) 573-9274.

SPEEDUP HOME ACCOUNTANT

This package speeds up Home Accountant file processing 4 to 60 times, searches up to 100 times faster, speeds up garbage collection 9 times. It also allows you to switch checkbooks/types with only one keystroke, prompts you when entering split transactions, records automatic transactions multiple times during the month, PLUS MORE. Dos 3.3 Home Accountant only. VISA/MC accepted. MONEYBACK GUARANTEE. Only $19.00 + $2.50 shipping/handling. SOFTWARE SOLUTIONS, 5516 Merritt Circle, Edina, MN 55436. For orders only; 1-800-835-2246 Ext 131.

LOTTER BOSS III — The sensible system

SUPERIOR lottery data management system. Includes: # generation, storage for: Lotto/3/4 digit winners. File editor, Screen dumps/Print-outs, Random field creator. On-screen tote boards: track, sort, tally compile & show status of data. Sort by occurrence, sift data for pairs/combos/etc. Menu driven & more. II+ ,e,c: $24.95 + $1.50 s/h. (ck/mo/visa/mc) Pa+6% 3 versions:A,B,C:Lotto 1-40,44,48. Please specify WALTER COMPUTER SERVICES, Box 97 Dept. NJ, Paradise, PA 17562 or call (717) 768-3006.

••• GENIE THE LOTTERY SPIRIT •••

Modify 3/4&6 digit numbers, Translate any word or phrase into numbers, Scramble numbers, & File numbers. Applesoft programming. Send a check or money order for $21.95 to **EPSOMIC, 470 Irwin, Pontiac, MI 48053**, 313-391-4297. Michigan residents add 4% for sales tax.

GAMES

DUNGEONS!

Special price on two popular text adventure games! For a limited time only MAROONED and ABDUCTED normally $39.95 ea. will sell for $13.95 ea. plus $1.50 for shipping. MAROONED begins deep in a dark dungeon and is filled with puzzles and chilling encounters with the inhabitants of this strange land. ABDUCTED is the sequel that takes you deep into Space to the mysterious planet of Albion. For Apple II+, IIe, and IIc only. VisionS, 610 West Alkaline Springs, Vandalia, Ohio 45377.

ADVENTURE SOLVES 1$ EACH; 8 FOR 5$!

INFOCOM-Zorks thru Cutthroats, SIERRA Hi-Res 0-6, Transyl, Quest, Cov Mirror, SAGAs 1-12, Mask Sun, Serp Star, all SIRIUS, Sherwood, Masq, Death Carib, Hulk, Ult 1-3 (hints), many more. Full list-SASE THEODORE CHEN, Box 37129-GT, Atlanta, GA 30332.

A DAY AT THE RACETRACK

Realistic simulation of horse-race handicapping. Database of over 100 horses is continually updated to provide ever-changing statistics. Over 100 performance elements per horse on 2 disks. Over a dozen different research and summary screens, allows 7 different types of bets. Animated HiRes race segments. Runs on any Apple II with 64K and one disk drive. Send $24.95 (check or M.O.) to: GERHARDT SOFTWARE, 32600 Concord Dept. 401, Madison Heights, MI 48071, (313) 589-3049.

THE LOST ARK OF THE COVENANT

An all new machine-language Hires adventure for the Apple II+/e/c with one disk drive. This two-diskette adventure contains over 150 Hires images created with Penguin Software's Graphic Magician. In your quest for the Lost Ark, you will explore a bizarre world full of wild beasts, tricks, traps and riddles. $19.95 (add $2 for hint sheets). ORDERS ARE SHIPPED IMMEDIATELY. Send check or M/O to: MICROSCAN, 1384 Lyon Pl/Dept N, Wantagh, NY 11793, (516) 826-5807.

SUPER STAR TREK

You've seen it on mainframes! All the thrills of the original Star Trek plus: Romulans equipped with cloaking devices, Klingon Commanders with tractor beams, defend starbases, launch probes, orbit planets, mine dilithium crystals, five play levels and more! Not copy protected! Only $19.95. SoftTrek Enterprises, 3646 Fieldcrest Dr., Dept. NB, Garland, TX 75042.

HARDWARE

Apple //c CP/M Plus System

THE BEST Apple CP/M system ever runs on Apple //c. The fastest clock, Z80H runs at 8MHz! The most powerful operating system, CP/M Plus by Digital Research, is included. Features 128K bank switched RAM, 12K printer spooler, keyboard buffer, mouse support, screen dump, auto login, HELP, DATE, and DEVICE. Compatible with thousands of free programs, no configuration required. $239. Precision Software, 6514 North Fresno Street, Milwaukee, Wisconsin 53224, (414) 353-1666.

AppleWorks $189! Apple IIc with monitor and stand $799! Apple IIc with color monitor and stand $999! Apple monitor II $119! Imagewriter II $499! MUCH MORE! Call or send for catalog 4A for details and latest prices! DISCOUNT COMPUTER ACCESSORIES, 445 NORTH PINE REEDSBURG, WI 53959, 608-524-2429.

SERIAL TO PARALLEL FOR THE //C
★★BUFFERED PRODUCTS★★

USE PARALLEL PRINTERS WITH YOUR APPLE IIC. ELIMINATE PRINTER DELAYS WITH OUR BUFFERED UNITS FOR SERIAL OR PARALLEL PRINTERS. UP TO 256K BYTES OF BUFFERING. CUSTOM DESIGNS AVAILABLE. PRICES START AT $79.95. TECHNOLOGIC SYSTEMS, PO. BOX 44, RAYMOND, OH 43067, 614/890-6960.

Prowriter™ — C. Itoh™ Printer Upgrade

Add Correspondence Quality to your 8510 / 1550. Our ROM replaces proportional font with no loss of speed. DIP switch or escape sequence selectable. SASE for print samples & installation terms. $40 for ROM & installation instructions, $45 New Style Prowriter with solid metal bottom cover plate. NY residents + 7%. Dealer inquiries invited. House Of Hardware, R.D. #1 Box 227, Burdett, NY 14818.

APPLE PERIPHERALS FACTORY DIRECT!

64K/80 Column Bd FOR IIe $39. Joystick II/IIe/IIc $15. 4/PORT game I/O $15. Paddles PR $15. Cooling Fan w/surg prot. $25. Mouse w/software II/IIe $49. TV Modulator $11. Add $3.00 shipping. COD +$2.75 Write/Call for comp. list. NEXO DISTRIBUTION, 8824 Golf Drive, Spring Valley, CA 92077, (619) 589-7928.

BUSINESS

Personal Computer Owners
CAN EARN $1000 to $5000
monthly selling simple services performed by their computer. Work at home in spare time. Get free list of 100 best services to offer. Write: C.I.L.C.I., PO. Box 60369, San Diego, CA 92106-8369.

EDUCATION

CHEM-WHIZ! for all chemistry students.

Includes interactive periodic table and the first MOLECULAR WEIGHT CALCULATOR! Finds molecular wt. of ANY molecule! Menu-driven. On-screen helps. Very user-friendly. Unprotected. Send $15.00 to TRANS-SYSTEM ENTERPRISES, PO. Box 32, Eagle, Alaska 99738 (907) 547-2273.

"GRAB BAG"

NEED TO LEARN BASIC?
TEACHING BASIC?
THEN THIS IS FOR YOU!

The BASIC learning Programs manual contains over 90 simple programs that teach fundamental concepts. Each lesson includes program listing, sample run, explanation and exercises. Effective for self-instruction or suitable as workbook or text for classroom use. In use nationwide by individuals, high schools, vocational schools and colleges. Versions available for IBM PC, Apple II family and TRS-80. Diskette available containing all programs in manual; may be copied for classroom use. Prices: Manual $8, diskette $25, including shipping. Please send your order to: Academic Computing Center, University of Wisconsin-River Falls, River Falls, WI 54022, (715) 425-3583.

APPLE /// IS FOR ME

We have formed a central clearing house to pool Apple /// software and hardware so that you, the devoted Apple /// User, can conveniently locate needed products. All items are sold at substantial discounts. To order current available product list send three dollars to: Association of Independent Microdealers (AIM), 3010 N. Sterling Av., Peoria, IL 61604.

AIM — APPLE II USERS GUIDE

This 300 page guide includes thousands of items for the Apple IIe-IIc computers which have been endorsed by the Association of Independent Microdealers (AIM). A detail listing of each product, manufacturer directory, association member directory, coupon section offering super values, handy index-price list, and several fun projects are included. This reference is updated regularly to insure current, up-to-date product listings. The guide is available individually for $16.95 or on a subscription basis for $75.00 for 12 full issues. To purchase the guide see your local AIM MEMBER or write: Association of Independent Microdealers, 3010 N. Sterling Av., Peoria, IL 61604.

••• $1000.00 REWARD •••

TO THE FIRST TO SOLVE MY PUZZLE
APPLE II+, E OR C
HURRY AND AIRMAIL $25 TO PUZZLESOFT,
BOX B82, HUDDERSFIELD UK, HD1 1XF

PROFESSIONALLY PRINTED

5 × 1½ inch label and quick reference guide set for APPLEWORKS, PRINT SHOP, CAPTAIN GOODNIGHT, COPY II+ and many more. Send $2.50 per set and .50 for shipping per order, money order or check only. To LABELS UNLIMITED, PO. BOX 2338, DAVENPORT, IOWA 52804, (319) 322-8555 for list.

APPLE CP/M Public Domain

UG Software Rental: CP/M UG Vol. 1-92 $45.00, SIG/M UG Vol. 1-230 $135.00, PICONET Vol. 1-34 $25.00, Pascal-Z UG Vol. 1-27 $25.00, UG Games 20 Vols of the best ones $25.00, UG Modem 20 Vols. of the best $25.00, UG Business 20 Vols. of the best $25.00, UG Utilities 10 Vols of the best $25.00. Rental is for 7 days after receipt with 3 more days grace for return. Credit cards accepted (preferred). Down-loading-disk format conversions. Call Soon! Apple DOS and MAC Librarys on Rent. 619-727-1015 24 hrs. 619-941-0925 info. PJ's National Public Domain Software Center, 1533 Avohill, Vista, CA 92083.

SAVE 30% or more on Apple and Mac software. Write for a Free catalog. Please specify your computer type. Delson's Catalog Offer, 19801 Vanowen St., Canoga Park, CA 91306, 818-887-5260.

NIBBLE LISTS

From time to time NIBBLE Magazine makes its subscriber mailing list available to reputable software and hardware manufacturers who wish to provide our readers with flyers, catalogs or other material of interest. All mailings are evaluated by NIBBLE Magazine to ensure the integrity and suitability of the material. Our policy forbids telephone or other direct solicitation by these firms, but does release your NIBBLE mailing label to them for their one-time use.

If you would prefer not to receive these mailings please send us a recent mailing label from the front of your magazine or copy the label exactly as it appears. Please send the label with a brief note directing us not to make your name available, and we will code your name for exclusion.

How to Renew your nibble® Subscription

Your NIBBLE Mailing Label contains important information for renewing your subscription.

	A	B
John Smith Anywhere St. Anytown USA 11111	0.111.C	8/86

A *Your* **SUBSCRIPTION REFERENCE NUMBER** should be included with your Renewal to assure rapid processing.

B Issue number and year with which your subscription will **expire**; your last issue will be delivered the month **before** this date. We must receive your renewal at least two months before this date in order for your subscription to continue unbroken.

Renew now and save $12.00 off the cover price!

(Use this coupon or the convenient **ORDER CARD** in this issue).

Please Enter My NIBBLE Renewal For:

☐ 12 Issues at $26.95 (U.S. only). ☐ 24 Issues at $49.95 (U.S. only).
☐ 36 Issues at $69.95 (U.S. only).

☐ Check/MO Enclosed Bill my ☐ VISA ☐ MasterCard

☐ Bill me (U.S. only).

☐ I'm an IAC member. Please renew my subscription and membership. I've enclosed my payment of $30.00 (U.S. only).

I use the following computer(s): ☐ Apple // Family ☐ Macintosh

Name _____ Ref. # _____

Address _____

City _____ State _____ Zip _____

Card # _____ Exp. _____
PLEASE PRINT CLEARLY

Signature _____

12 issue subscription rates:
U.S. Air Mail — $51.95
Canada Surface — $34.95
Canada Air Mail — $59.95
Foreign Surface — $39.95
Foreign Air Mail — $89.95

Mail To: NIBBLE
45 Winthrop St.
Concord, MA 01742

Six Years of Nibble At Your Fingertips

THE NIBBLE INDEX

The Nibble Index is the reader's quide to everything that has been published in Nibble from 1980 to 1985, **on disk**! It covers articles, letters, and errata with **more than 1,000 entries**!

Highlights:

- Fast machine language search
- Search on any keyword to find an author, title, or subject
- Send search results to the screen or printer
- Utilities let you print out all data sorted by title or author and convert data to text file format for use in your word processor
- Also includes data in AppleWorks database format*

Now you can track your favorite author . . . find useful utilities . . . trace the enhancements to your favorite program . . . just about everything!

Whether you're a newcomer to Nibble or a long-time reader, the Nibble Index will make "Nibbling" a whole lot easier!

*To use this data you must have AppleWorks and 128K.

To order fill out the attached coupon or call 617-371-1660.

I want the Nibble Index! Here's my $14.95

☐ Check, M.O. Postpaid

(Payable in U.S. funds only; Mass. residents add 5% sales tax)

Name _____

Address _____

City _____ State _____ Zip _____

☐ MasterCard ☐ Visa

Signature _____ Tel. # _____

Charge Card # _____ Exp. Date _____

nibble 45 Winthrop St., Concord, MA 01742 (617) 371-1660

ONE-LINER AND TWO-LINER WINNERS

This one-liner by Andrea Sigurdson of Sydney, BC, Canada is a cute way to display a message. Just follow the arrow as it points to each letter. To program your own message, replace the DATA elements at the end of the line with your own values. A is represented by 1, Z by 26, and the separation between words by 33. Then change the FOR A statement to the total number of DATA elements.

```
5  HOME : PRINT "WATCH THE
     ARROW SPELL A MESSAG
     E": VTAB 8: PRINT "AB
     CDEFGHIJKLMNOPQRSTUVW
     XYZ NEW WORD": FOR A =
     1 TO 31: VTAB 9: READ
     H: HTAB H: PRINT "^";
     : FOR C = 1 TO 2000: NEXT
     C: HTAB H: PRINT " ":
      NEXT A: DATA  20,1,1
     1,5,33,1,33,14,9,2,2,
     12,5,33,15,6,33,1,14,
     33,1,16,16,12,5,33,20
     ,15,4,1,25
```

With this DOS 3.3 utility by Eric Snyder of Bloomington, MN, you can display the disk catalog and then choose from a menu of ten DOS commands. The program then asks for the file name and executes the command. For the commands PR# and IN#, enter the slot number when the file name is requested.

```
2  PRINT  CHR$ (4);"CATALO
     G": RESTORE : FOR A =
     1 TO 10: READ C$(A): NEXT
     A: FOR A = 1 TO 10: PRINT
     A;"=";C$(A): NEXT A: PRINT
     "11=CATALOG": INPUT "
     CHOICE? ";A: ON (A <
     1 OR A > 10) GOTO 2: INPUT
     "FILE? ";B$: PRINT  CHR$
     (4);C$(A);" ";B$: DATA
     LOAD,BLOAD,RUN,BRUN,U
     NLOCK,LOCK,DELETE,VER
     IFY,PR#,IN#
```

Here's a DOS 3.3 utility that modifies the CATALOG command. When you execute the CATALOG command, the program asks for the file type (A, B, I, R, S, T or =), and then lists only that type of file. Thanks to Michiel Berger of Den Burg Texel, Holland.

```
1 D$ =  CHR$ (4): PRINT D$
    "OPENCAT": PRINT D$"W
    RITECAT": PRINT "CALL
    -151": PRINT "ADAF:8
    BD B0 N ADB9:20 C FD
    85 0 20 ED FD 20 2F A
    E 20 2F AE EA EA N AD
    DD:4C E8 AD N ADF6:20
     B8 B6 N B6B8:B9 A7 B
    3 C5 0 D0 1 60 A5 0 C
    9 BD F0 6 B9 A7 B3 4C
     25 AE B9 A7 B3 60"
2  PRINT "B3B0:BA C5 D0 D9
     D4 C5 CC C9 C6": PRINT
    "3D0G": PRINT D$"CLOS
    ECAT": PRINT D$"EXECC
    AT"
```

Gene Masse of Belchertown, MA sent us an eye-catcher — his two-liner simulates the motion of a bouncing ball. When you run it, a ball bounces up and down on the Hi-Res screen until it comes to rest.

```
1  FOR G = 24576 TO 24582:
     READ R: POKE G,R: NEXT
     : POKE 232,0: POKE 23
    3,96:X = 140:Y = 0:M =
    0: HGR2 : SCALE= 2: ROT=
    6: FOR G = 1 TO 180: HCOLOR=
    3: DRAW 1 AT X,Y: IF
    Y > 180 THEN M =  - M
2  DRAW 1 AT X,Y:M = M + 2
     : HCOLOR= 0: DRAW 1 AT
    X,Y:Y = Y + M: FOR P =
    1 TO 20: NEXT P,G: HCOLOR=
    3: DRAW 1 AT X,Y: DATA
    1,,4,,53,39,
```

Many variations on John Conway's game of Life have been created. In this two-liner version by William K. Richards of Martinez, CA, each dot represents a living cell. If a living cell is surrounded by two or three living cells, it survives to the next generation. A dead cell comes to life when it is surrounded by three living cells. This Hi-Res version is seeded by an r-shaped pentomino pattern. Just BRUN LIFE or BLOAD LIFE and CALL 4096.

To seed it with your own starting pattern, start by performing HGR and HGR2. Then POKE the location of each living cell into memory using the formula:

POKE X * 256 + 8192 + 128 + Y, 255

where X and Y are the screen coordinates in the range 1-38. CALL 4119 to start the manually seeded version. To key in the program, enter the Monitor with CALL −151, type in the hex code, and save it with the command:

BSAVE LIFE,A$1000,L$B0

```
1000- 20 D8 F3 20 E2 F3 A9 FF
1008- 8D 95 34 8D 95 35 8D 96
1010- 34 8D 96 33 8D 97 34 AD
1018- 50 C0 AD 54 C0 AD 56 C0
1020- 20 36 F8 A9 00 85 06 A9
1028- 80 85 08 A2 27 86 EB A0
1030- 27 84 EC A5 EB 18 69 20
1038- 85 07 85 09 A4 EC B1 08
1040- 85 30 29 01 91 06 A4 EB
1048- A5 EC 20 00 F8 C6 EC D0
1050- EB C6 EB D0 DA A2 27 86
1058- EB A0 27 84 EC A5 EB 18
1060- 69 20 85 07 85 09 A4 EC
1068- A9 00 18 71 06 C8 71 06
1070- C6 07 71 06 88 71 06 88
1078- 71 06 E6 07 71 06 E6 07
1080- 71 06 C8 71 06 C8 71 06
1088- 88 C6 07 C9 03 F0 0E C9
1090- 04 F0 06 A9 00 91 08 F0
1098- 08 B1 06 F0 F6 A9 FF 91
10A0- 08 C6 EC D0 C1 A0 27 84
10A8- EC C6 EB D0 B0 4C 2B 10
```

TYPING TIPS

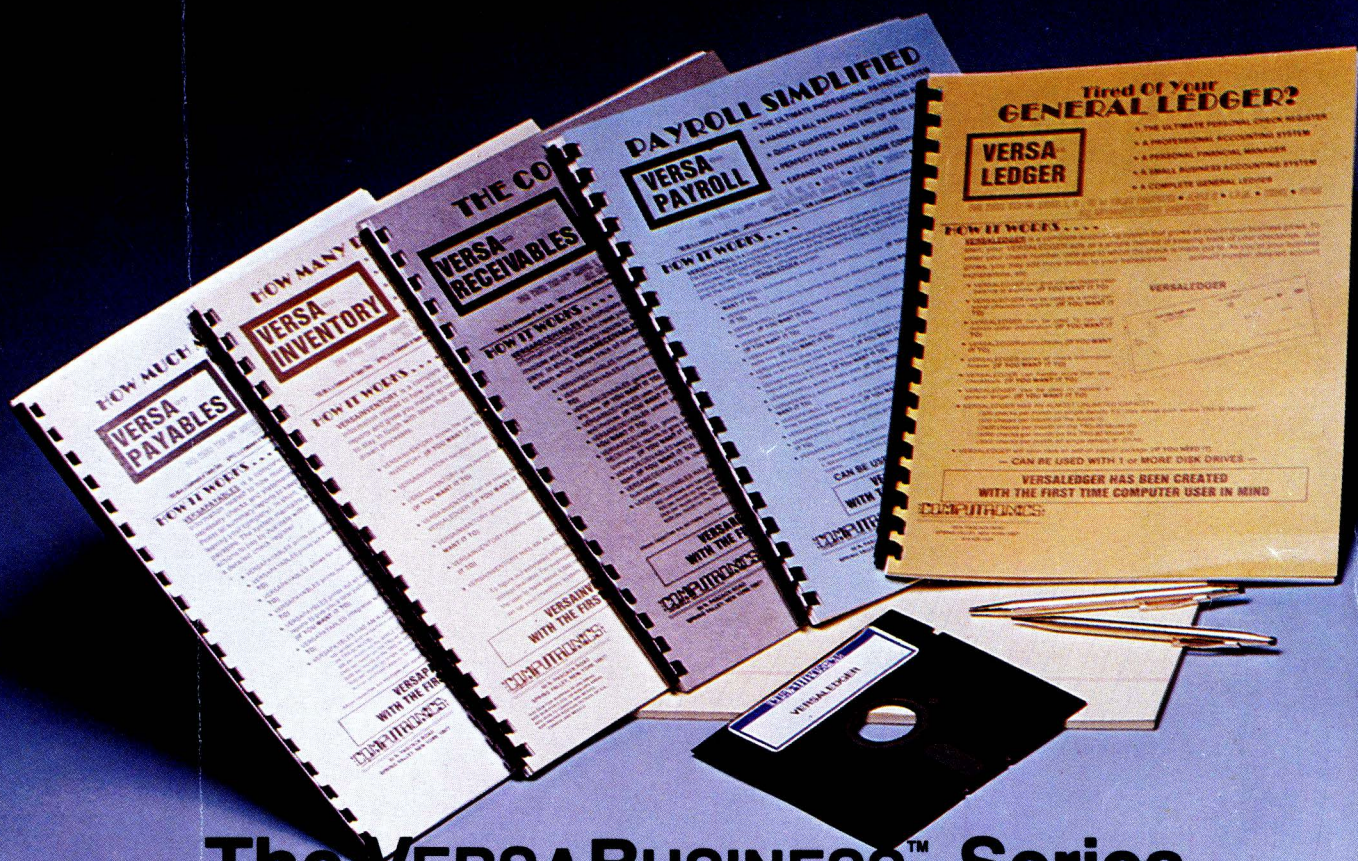
Here are a few tips for typing in one-liners and two-liners. Be sure to omit spaces when typing in Applesoft programs, unless the spaces appear in strings between quotes, or in REM or DATA statements. You'll find it easier (and often essential) to substitute the question mark (?) for the PRINT statement. And when typing in machine language programs, it's safe to type in just the second digit of a hexadecimal pair if the first digit is a zero, e.g., the hex pair 03 can be typed in simply as 3. For more information on typing in programs, see "A Welcome to New *Nibble* Readers" at the beginning of this issue.

THE GROUND RULES

The One-Liner and Two-Liner Contests are ongoing events with no deadlines. Several winners are chosen each month, and if you win, your prize will be the *Nibble* disk of your choice. The challenge is for you to create the most interesting, attractive, useful, and/or clever program that can be typed using just one or two program lines. The lines must be enterable directly from the keyboard, without using a programming aid or line editor. Programs must be your own, original work, and may be written in Applesoft, machine language, or any of the languages available for the Macintosh. Programs must be submitted on disk, accompanied by an 8 1/2 by 11 inch sheet of paper with your name, address, *Nibble* disk choice, and the words "One-Liner Contest" or "Two-Liner Contest." We're sorry, but we are unable to return your entry disk. All entries become the property of MicroSPARC, Inc.

Introducing the Most Powerful Business Software Ever!

TRS-80™ (Model I, II, III, or 16) • APPLE™ • IBM™ • KAYPRO™ • CP/M™ • COMMODORE 64™



The VERSABUSINESS™ Series

Each VERSABUSINESS module can be purchased and used independently, or can be linked in any combination to form a complete, coordinated business system.

VERSARECEIVABLES™ $99.95

VERSARECEIVABLES™ is a complete menu-driven accounts receivable, invoicing, and monthly statement-generating system. It keeps track of all information related to who owes you or your company money, and can provide automatic billing for past due accounts. VERSARECEIVABLES™ prints all necessary statements, invoices, and summary reports and can be linked with VERSALEDGER II™ and VERSAINVENTORY™.

VERSAPAYABLES™ $99.95

VERSAPAYABLES™ is designed to keep track of current and aged payables, keeping you in touch with all information regarding how much money your company owes, and to whom. VERSAPAYABLES™ maintains a complete record on each vendor, prints checks, check registers, vouchers, transaction reports, aged payables reports, vendor reports, and more. With VERSAPAYABLES™, you can even let your computer automatically select which vouchers are to be paid.

VERSAPAYROLL™ $99.95

VERSAPAYROLL™ is a powerful and sophisticated, but easy to use payroll system that keeps track of all government-required payroll information. Complete employee records are maintained, and all necessary payroll calculations are performed automatically, with totals displayed on screen for operator approval. A payroll can be run totally, automatically, or the operator can intervene to prevent a check from being printed, or to alter information on it. If desired, totals may be posted to the VERSALEDGER II™ system.

VERSAINVENTORY™ $99.95

VERSAINVENTORY™ is a complete inventory control system that gives you instant access to data on any item. VERSAINVENTORY™ keeps track of all information related to what items are in stock, out of stock, on backorder, etc., stores sales and pricing data, alerts you when an item falls below a preset reorder point, and allows you to enter and print invoices directly or to link with the VERSARECEIVABLES™ system. VERSAINVENTORY™ prints all needed inventory listings, reports of items below reorder point, inventory value reports, period and year-to-date sales reports, price lists, inventory checklists, etc.

VERSALEDGER II™ $149.95

VERSALEDGER II™ is a complete accounting system that grows as your business grows. VERSALEDGER II™ can be used as a simple personal checkbook register, expanded to a small business bookkeeping system or developed into a large corporate general ledger system **without any additional software.**

- VERSALEDGER II™ gives you almost unlimited storage capacity (300 to 10,000 entries per month, depending on the system),
- stores all check and general ledger information forever,
- prints tractor-feed checks,
- handles multiple checkbooks and general ledgers,
- prints 17 customized accounting reports including check registers, balance sheets, income statements, transaction reports, account listings, etc.

VERSALEDGER II™ comes with a professionally-written 160 page manual designed for first-time users. The VERSALEDGER II™ manual will help you become quickly familiar with VERSALEDGER II™, using complete sample data files supplied on diskette and more than 50 pages of sample printouts.

SATISFACTION GUARANTEED!

Every **VERSABUSINESS™** module is guaranteed to outperform all other competitive systems, and at a fraction of their cost. If you are not satisfied with any **VERSABUSINESS™** module, you may return it within 30 days for a refund. Manuals for any **VERSABUSINESS™** module may be purchased for $25 each, credited toward a later purchase of that module.
All CP/M-based Computers must be equipped with Microsoft BASIC (MBASIC or BASIC-80)

To Order:
Write or call Toll-free (800) 431-2818
(N.Y.S. residents call 914-425-1535)

* add $3 for shipping in UPS areas
* add $4 for C.O.D. or non-UPS areas
* add $5 to CANADA or MEXICO
* add proper postage elsewhere

VISA MASTERCARD

DEALER INQUIRIES WELCOME

All prices and specifications subject to change / Delivery subject to availability.

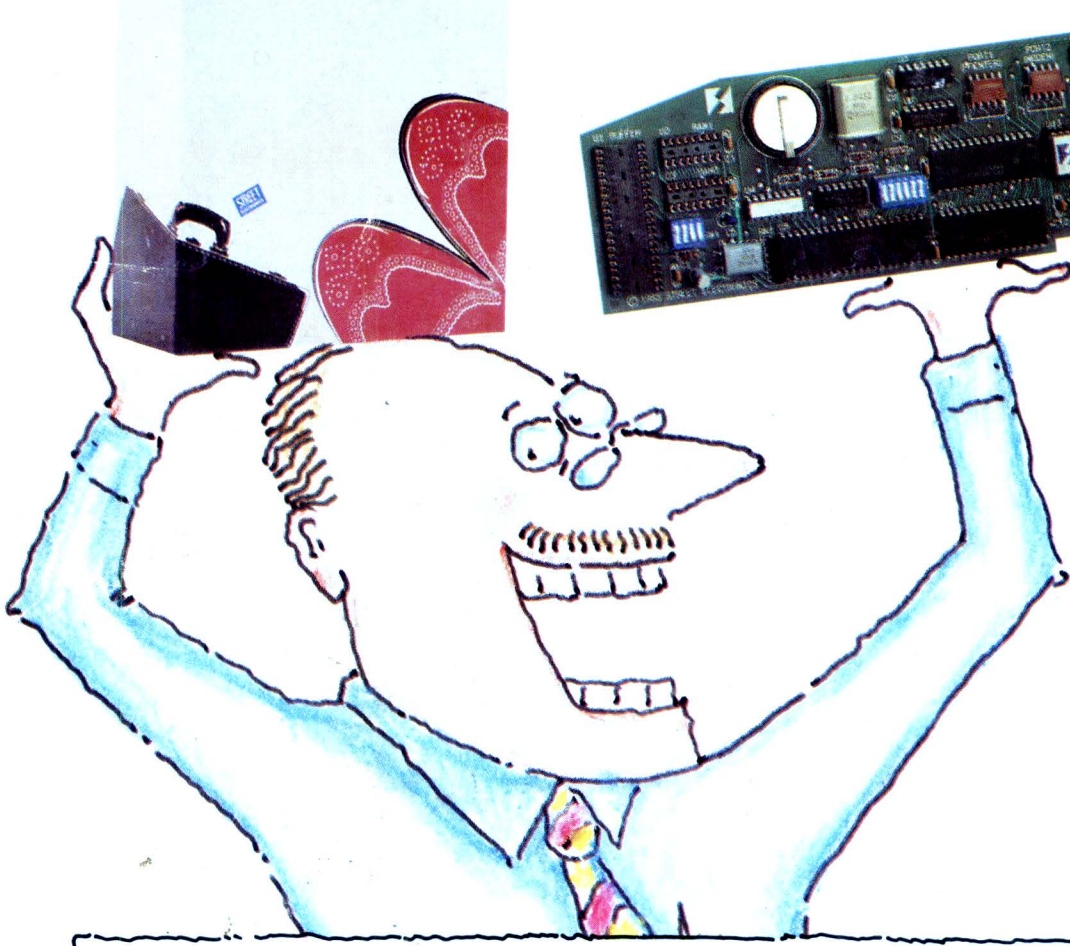
COMPUTRONICS INC

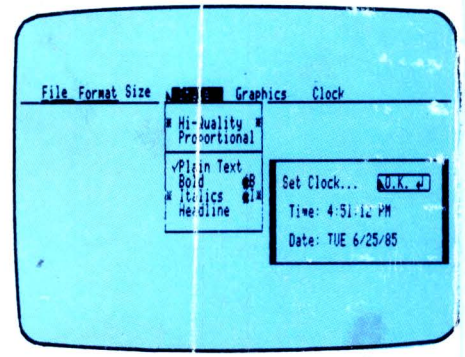
50 N. PASCACK ROAD, SPRING VALLEY, N.Y. 10977

Street Electronics'

BusinessCard™

Intelligent Multifunction Interface for the Apple™ IIe





BusinessCard Menu

Product Features:

- Serial printer port (parallel also avialable)
- Modem port
- Pro-DOS compatible clock/calendar with a battery
- Extensive built-in text and graphics commands *(Includes selectable text size and style; graphic windowing and magnification)*
- Supports new Apple color ImageWriter™ II printer
- Easy-to-use "Macintosh™ like" pull-down menus *(For selecting text and graphics features)*
- Prints double HI-RES color graphics
- Appleworks and Mouse compatible
- Super Serial Card and Apple IIc compatible hardware design
- Optional print buffer ($79.95 for 64K)

Cost Comparison

	IF PURCHASED SEPARATE	*BUSINESSCARD*
Printer Card	$139.95	Included
Modem Card	$139.95	Included
Clock Card	$149.95	Included
Screen Printing Program	$ 49.95	Built-In
Total	**$479.80**	**$219.95**

Send for our brochure.

Street Electronics Corporation
1140 Mark Avenue
Carpinteria, CA 93013
(805) 684-4593